2008-09-22

Development of a Workflow for the Comparison of Classification Techniques

Zanifa Omary

Technological University Dublin, zanifa.omary@student.dit.ie

Development of a Workflow for the Comparison of Classification Techniques

Zanifa Omary
School of Computing
Dublin Institute of Technology
Kevin St. Dublin 8, Ireland.

Msc Computing
(Information Technology)

Dublin Institute of Technology, School of Computing
September 2008

Declaration

I hereby certify that this dissertation which I now submit for assessment by the School of Computing, Dublin Institute of Technology on the programme of study leading to the award of ***Msc Computing (Information Technology)*** is entirely my own work and has not been submitted for assessment for any academic purpose other than in particular fulfilment for the stated above.

Signed

_____

Zanifa Omary

Date

22nd September 2008

Abstract

As the interest in machine learning and data mining springs up, the problem of how to assess learning algorithms and compare classifiers become more pressing. This has been associated with the lack of comprehensive and complete workflow depending on the project scale to provide guidance to its users. This means the success or failure of the project can be highly dependent on the person or team carrying it.

The standard practice adopted by many researchers and experimenters has been to follow steps or phases from existing workflows such as CRISP-DM, KDD and SAS-SEMMA. However, as machine learning and data mining fields involve complex comparative experiments, there is a need of having complete workflow which when applied provides efficient and effective results. Though existing workflows offers many benefits, a successful comparative experiment requires more than outlined steps of workflows. Conclusions based on results drawn from a more complete workflow will yield more reliable results and experimenter can stand with confidence while comparing classifiers.

This dissertation focuses on a range of issues from machine learning to statistics for the development of the classifier workflow. It represents in detail background materials which are the key to understanding how different experiments have to be carried out. It explains how different classification techniques work and their applications in different areas. It also explains how classification evaluations can be used in different domains. It also determines when an experimenter should use performance measures and how these measures correspond to performance estimators. Moreover, it explains how different settings can be obtained before committing to the experimentation step. Finally, a complete eight-phase classifier workflow which is platform independent will be provided. The workflow was then evaluated by expert users using close ended questionnaire.

Keywords: *Machine learning, supervised learning, performance measures, performance estimation method, classifier workflow, parameter settings, classifier, classification techniques, threshold.*

Acknowledgements

Foremost, I am deeply indebted to my supervisor, Dr. Brian Mac Namee of the School of Computing, Dublin Institute of Technology for many insightful conversations during the development of ideas of this thesis. Some stimulating suggestions, constructive criticisms and encouragement helped me in all the time of research.

I would also like to thank Dr. Ronan Fitzpatrick for his interesting and challenging seminars during the taught part of this course. He has always been more than willing to answer my naïve questions in the preparation of this dissertation and acted as a good friend along the way.

I would also like to thank Dr. Fred J. Mtenzi for his never ending encouragement and support from day one in Ireland. Throughout this course and especially during the project time he has been a source of encouragement and support of all kinds.

My parents have been an inspiration throughout my life. They have always supported my dreams and aspirations, and if I do say so myself, I think they did a fine job raising me. I would like to thank them for what they are, and all they have done for me.

Table of Contents

Table of Figures

Table of Tables

Glossary of Acronyms Used

ANN	Artificial Neural Network
AUC	Area under ROC Curve
AUC-ROC	Area Under Curve-ROC
CART	Classification And Regression Tree
CRISP-DM	Cross Industry Standard Process for Data Mining
Cv/CV	Cross Validation
DOE	Design of Experiments
DT	Decision Tree
EM	Enterprise Miner
FN	False Negatives
FP	False Positives
ID3	Iterative Dichotomiser 3
IR	Information Retrieval
ISL	Integral Solution Limited
KDD	Knowledge Discovery in Databases
KNN	K-Nearest Neighbour
ML	Machine Learning
NCR	National Cash Registers
NoF	Number of Features
OCR	Optical Character Recognition
PCA	Principal Component Analysis
RBF	Radio Basis Function
RF	Random Forest
ROC	Receiver Operating Characteristics
SLN	Single Layer Network
SVM	Support Vector Machines
TN	True Negatives
TNR	True Negative Rate
TP	True Positives
TPR	True Positive Rate
UCI	University of California, Irvine

1 Introduction

The aim of this chapter is to provide an introduction of the dissertation. Section 1.1 provides an overview of the dissertation followed by the discussion on research problem in section 2.2. The main aim and objectives of the research have been explained in section 1.3. Finally section 1.4 describes the structure and organisation of the dissertation.

1.1 Overview of the research project

For long the construction of machines that are capable of learning from experience has been seen as an objective of many fields ranging from financial to health institutions. This development was nearly impossible when the processing speed of computers was very low. The advent of computers, probabilistic frameworks and information age, where voluminous of data are generated from day-to-day activities has attracted different fields to incorporate machine learning and its related classification techniques to solve problems existing in such fields (Baldi & Brunak 2001). When a powerful computer is trained to perform certain task such as predicting credit risk in financial institution, this is what we refer to as machine learning. In literature review classification research, which is a component of data mining and a subfield of machine learning, has been identified to provide the interaction between the two research areas; machine learning and data mining (Salzberg 1999).

Classification research which includes methodologies for comparing the performance of classifiers in a single application domain is considered as an understudied area (Prechelt 1996; Salzberg 1999). Salzberg (1999) proposes a need of very specific and focused studies while comparing classifiers or assessing the performance of learning algorithms in any domain of engineering. The evidence to this need has been provided by Prechelt (1996) which indicates that the evaluations to classification research are not done nearly enough and result into serious experimental deficiencies and even making statistically invalid conclusions. Classification research comes in a variety of forms: as it lays out new algorithms and shows their feasibility in real world application domains or describes creative new algorithms which sometimes do not require experimental validation (Salzberg 1999). From significances that

classification research has to machine learning and data mining, Salzberg recommended a need of having a statistically acceptable framework for its comparative work.

Salzberg (1999) proposes a framework for the comparison of classification techniques which consists of data repositories, new algorithms and comparative study and proper methodology. The framework was intended for those who are already familiar with machine learning and data mining experiments as it suggests pitfalls that an experimenter has to avoid while comparing classifiers. The aim of this dissertation is to develop the classifier workflow for the comparison of classification techniques in a single domain. Salzberg's framework did not propose anything related to non-expert users of the framework, application domain and a number of steps that will provide improved performance measure. From identified weaknesses, eight phased classifier workflow have been proposed by the author for machine learning and data mining classification projects for single application domain.

The developed classifier workflow has been characterised as being applicable to small scale classification projects, provide guidelines on how different phases can be performed ranging from the selection of which statistical test to use while comparing two-dimensional classifiers, how to find the parameters of the algorithms that are going to be compared and used in such application domain and how to set the threshold for performance estimation methods. Therefore throughout this dissertation, threshold for performance estimation method will be regarded as the maximum number of instances that should be considered applicable for a performance estimation method.

1.2 Research problem

The principal aim of the research described in this dissertation is to develop a workflow for the comparison of classification techniques for a particular application domain. From the existing research literature in data mining there are a number of workflows or refered as life cycles by many researchers that are mostly used while creating models or comparing performance between classifiers for large scale projects. However, these workflows lack clear procedural steps and tasks for

experimenters to follow while creating models from certain algorithms or comparing performance of the classifiers in small scale projects in different application domains. This has resulted in time consuming and repetitive nature of the tasks as well as it becomes easy for the experimenter to make statistically invalid conclusions.

To develop the classifier workflow, it was necessary to perform an extensive literature review around the machine learning field in order to gain an understanding of machine learning and its applications and how classification techniques and existing workflows contribute to this and evaluated. The literature review was divided into two parts, the first part provided background in machine learning where the focus was on a definition of machine learning, examples of machine learning applications, categories of machine learning followed by an overview of classification techniques and existing workflows. The second part dealt with different classification evaluation schemes. These two parts were used as inputs to the proposed classifier workflow provided in chapter 4. Experiments in chapter 5 aimed at establishing benchmark settings for the proposed classifier workflow.

1.3 Research objectives

The following objectives have been achieved throughout the dissertation and contributed to the overall outcome of the dissertation

1. To investigate and explore machine learning and techniques used while creating models for small scale projects.

2. To explore and evaluate a number of workflows and qualify their phases for small scale machine learning and data mining projects. The analysis of the existing workflows will provide inputs to the proposed classifier workflow.

3. To investigate classification evaluation schemes. The evaluation is categorised into three parts, performance measures, statistical tests and performance estimation methods. The performance measures such as accuracy, precision and recall are used to measure the performance of classification techniques while statistical tests are used for assessing the

significance of the difference between learning algorithms. The performance estimation methods will be used to estimate the performance of the classifiers produced.

4. To perform experiments for setting up unknown settings of the proposed classifier workflow.

1.4 Organisation of the dissertation

The remaining chapters of this dissertation are organised as follows:

- Chapter 2 gives some theoretical background to the field of machine learning. The chapter begins with an overview of machine learning in general, in order to provide an introduction to the subject for those who are unfamiliar with it. Later discussion is based upon examples of machine learning applications in different fields. Categories of machine learning are provided in the second part of the chapter. The third part focuses on classification techniques such as decision trees, neural networks, k-nearest neighbour, support vector machines and random forests. The last part of the chapter provides an overview of existing workflows in machine learning.

- Chapter 3 discusses techniques for evaluating the performance of classifiers. The chapter is divided into five section; introduction, performance measures, statistical tests and their evaluation, and performance estimation methods. With performance measures focus will be on accuracy, precision, recall, f1-score and ROC analysis. McNemar's test, a test for the difference of the two proportions, resampled paired t test, k-fold cross validated and 5 x 2 cross validated paired t test are the statistical tests considered for assessing and comparing classification algorithms discussed in the third section of the chapter. Statistical test evaluation is presented on the fourth section. The last section provides three performance estimation methods; holdout test method, k-fold cross validation and leave-one-out method.

- Chapter 4 provides proposed classifier workflow for the comparison of classification techniques in a single application domain. The chapter will be introduced before the presentation of the proposed classifier workflow. The third section provides a discussion on the eight phases of the workflow; experimental design, algorithm selection, preprocessing, performance estimation method and performance measures. Other phases are algorithm parameters, experimentation and evaluation followed by the unknown settings from the proposed classifier workflow. These unknown settings require experiments which will be performed in chapter 5 for its full use and application to the complete classifier workflow.

- Chapter 5 introduces experiments that will be used for establishing benchmark settings outlined in chapter 4.Three different experiments were conducted, setting the threshold of the dataset, deciding on the usage of each performance estimation method and setting up one or two key parameters for the classification technique that will be used for the experiments in an application domain. Different datasets with different dimensionality will be used as inputs to the experiments associated with such an application domain. Classifiers will be tested using not more than two key parameters together with three parameter values.

- Chapter 6 provides the final and complete classifier workflow based on the experiments performed in chapter 5. The chapter has been divided into four sections. The chapter is introduced before presentation of the results from the evaluation survey. The second section provides the results which were obtained from the classification workflow evaluation survey. The recommendations from the survey are also outlined in this chapter. The third section provides walkthrough of the classifier workflow before conclusion of the chapter in the fourth section.

- The seventh and final chapter of this dissertation summarises the research, provides conclusions and discusses areas identified for further research.

2 Background

2.1 Introduction

In recent years, the goal of many researchers' in different fields is to build systems that can learn from experience and adapt to their environments. This evolution has resulted in various algorithms that are transforming problems rising from industrial and scientific fields. Machine learning, as a field which comprise of tools for building models, has resulted in the convergence of several communities from different fields such as statistics, artificial intelligence, philosophy, information theory, cognitive science and biology for the sake of solving problems arising into those fields.

This chapter provides the background material for the remainder of this dissertation. The first section provides the definition of machine learning followed by examples of machine learning applications. The third section introduces the two key categories of machine learning; supervised and unsupervised learning. The fourth section provides an overview of five popular classification techniques. These materials are relevant since the proposed workflow presented in chapter four depends on concepts provided in this chapter. The last section is an overview of the existing workflows in data mining that are mostly used.

2.2 What is machine learning?

Prior to delving into formal definitions of machine learning it is worthwhile to define two terms that make-up machine learning; machine and learning. In information technology context machine is a device, especially computer that accepts data manipulates them and produces output information based on a sequence of instructions on how the data has to be processed. On the other hand, learning is the process of acquiring skills or knowledge. Therefore, a complete definition of machine learning for this dissertation has to incorporate the computer based knowledge acquisition process and has to state where skills or knowledge can be obtained.

Mitchell (1997) defines machine learning as *"a field which deals with the issue of how to build computer programs that use experience from past tasks to improve their*

performance". This definition does not reflect anything related to knowledge acquisition process for the stated computer programs, therefore it is considered incomplete for this dissertation.

Alpaydin (2004) provides a more formal definition of machine learning. Alpaydin defines machine learning as "*the capability of the computer program to acquire or develop new knowledge or skills from existing or non existing examples for the sake of optimising performance criterion"*. This definition is more preferred as it directly correlates with the principal aim of this research problem and it incorporates knowledge in its definition which the former definition did not include.

The growing interest in machine learning is driven by two factors as outlined by Alpaydin (2004), *removing tedious human work* and *reducing cost*. As the result of automation of processes, huge amounts of data are produced in our day-to-day activities. Doing manual analysis on all of this data is slow, costly and people who are able to do such analysis manually are rare to be found (Fayyad, Piatetsky-Shapiro & Smyth 1996). Machine leaning techniques when applied to different fields; finance, manufacturing, telecommunication and medicine (Alpaydin 2004) have proved to work with huge amounts of data and provide results in a matter of seconds.

Machine learning has been widely related to knowledge engineering process which is a field within artificial intelligence (AI) that develops knowledge-based systems (Langley & Simon 1995). Knowledge engineering process involves the process of integrating knowledge from expert(s) or expert sources into computer systems (Sebastiani 2002). Such systems are computer programs that contain large amounts of knowledge, rules and reasoning mechanisms to provide solutions to real-world problems. As defined previously, machine learning involves the process of acquiring knowledge from existing and non-existing samples which does not involve experts or expert sources. Rather it uses existing or non-existing examples from training data to develop or acquire new knowledge and skills (Langley & Simon 1995).

Examples of machine learning applications

As related to the relationship between machine learning and knowledge engineering process provided in section 2.2, the principal aim of machine learning is to increase the level of automation in the knowledge engineering process replacing time-consuming human activities with automatic techniques that improve efficiency by discovering regularities in training data (Langley & Simon 1995; Alpaydin 2004). Although machine learning has a wide continuum of applications ranging from learning to medical diagnosis to learning to assess credit risks of loan applicants, for this dissertation only three examples will be considered. The remainder of this section will discuss a number of such examples.

2.2.1 Pattern recognition

In our day to day lives, we are able to recognise countless patterns without any knowledge of how this happens. Consider as an example recognising your relatives' faces, as in figure 1 where by each person's face is associated with several characteristics such as the position of mouth, nose and eyes located in certain places of the face. Each person's face is a pattern composed of a particular combination of these features (Alpaydin 2004). Just like ourselves, learning algorithms have the ability of capturing the pattern specific to such person and then analyse sample face images and then recognise by checking for this pattern in a given image.

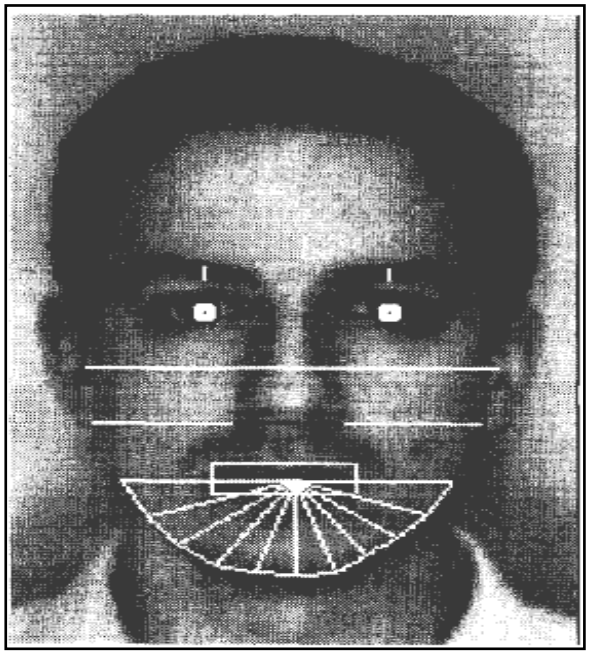

Figure 1: Patterns for the facial recognition experiments
(Source: (Brunelli & Poggio 1993))

Another instance of pattern recognition is Optical Character Recognition (OCR) which involves identification of characters in a document page (Rice, Nagy & Nartker 1999). Raynor (1999) defines OCR as the process of *"converting the image of an item containing text into character representation of that image"*. People have different handwriting styles, characters may be written small or large, with a pencil or pen, and there may be many images corresponding to the same character. Learning algorithms have the ability to recognise the same character despite having several differences (Rice, Nagy & Nartker 1999).

2.2.2 Credit application

When looking for details before providing loans to customers, financial institutions and loan companies use application forms or questionnaires to collect information about customers applying for the loans (Carter & Catlett 1987). The information collected may include income, savings, profession, age, past financial history and gross annual income (Alpaydin 2004). From the collected data, financial institutions aim to infer the general rule coding the association between customer's attributes and associated risk level (Galindo & Tamayo, 2000). It is important for the financial institutions and loan companies to predict in advance if the customer is likely to repay

the loan on time or not as these institutions and companies are inevitably exposed to the element of risk.

Financial institutions use machine learning techniques however, to decide whether to give the loan or not to customers depending on a threshold that has been determined by an institution. Threshold in credit application refers to minimum preconditions that must be met before the loan is issued to the applicant. If the customer has reached certain threshold for example, the bank may accept the application and if fell below threshold then the application is rejected. Most of the financial institutions such as American Express, UK (https://home.americanexpress.com/home/uk) use machine learning methods such as decision tree (CART) classifiers to make predictions on the risks associated with respective customers (Langley & Simon 1995). Other machine learning techniques for credit applications include neural networks, k-nearest neighbour, logit analysis and support vector machines (Mramor & Zupan 2008).

2.2.3 Medical diagnosis

In medicine, diagnosis has been defined as "*determination of the nature of the diseased condition; identification of a disease by careful investigation of its symptoms and history; also, the opinion resulting from such investigation"* (Oxford English Dictionary 1989). In last few years, the digital revolution has provided inexpensive and relatively easy ways for collecting and storing data in modern hospitals (Kononenko 2001). These modern hospitals have been well equipped with data collection and monitoring devices. A good example is the Software Wedge Universal RS-232 Data Collection Software from ColeParmer (www.coleparmer.com) which is used for data collection and stores them directly to spreadsheets, databases and statistical packages.

These tools store and share collected data using large information systems. Data about correct diagnoses are often kept in terms of medical records in their specialised departments or hospitals (Kononenko 2001). All that needs to be done during diagnosing is input patients' records or symptoms into a computer program, then run the trained learning algorithm. Machine learning provides learning algorithms that have been trained using patients' symptoms or records for diagnosing new patients. Despite all these developments, when it comes to *life* and *death* decision, automated

diagnosis for some diseases such as colon cancer and stroke for example, are still the major research problems in medical diagnosis. This has resulted from uncertainty and difficult decision in the selection of most personalised treatment option (Siddiqi et al. 2008)

2.3 Machine learning categories

Machine learning provides two important learning categories, namely supervised and unsupervised learning. These two learning categories are associated with different machine learning techniques that represent how the learning method works. In this section, two core learning categories; supervised and unsupervised learning are discussed. These materials serve as an introduction to the next section of this chapter where classification techniques are discussed. More discussion will be based on supervised learning as the aim of this dissertation is to develop a classifier workflow which falls under the supervised learning category.

2.3.1 Supervised machine learning

With supervised learning there is a presence of the outcome variable to guide the learning process (Hastie, Tibshirani & Friedman 2001). There is a variety of supervised learning methods such as decision trees, neural networks, KNN, SVM and random forests which attempt to discover the relationship between the input variables and the class attribute (Rokach & Maimon 2005). Given some training data described in terms of a set of features and their class labels, consider table 1, the goal of supervised learning is to find the partitioning of the attributes that allow correct classification of the training data as well as generalisation from training data to unseen, similar data (Caelli & Bischof 1997). Due to the presence of predefined examples or classes while learning, supervised learning is also refered to as classification (Ziarko & Yao 2001).

In a very clear way, with supervised learning, a *"supervisor"* is present to indicate if the system performs correctly or incorrectly, if the desired response from the system has been achieved or not, to validate the acceptability of the system's response, or to indicate the amount of error in system performance (Mehrotra, Mohan & Ranka 1997). The discovered relationship between inputs and outputs is represented in a

structure known as a *model* (Rokach & Maimon 2005). Models are usually used for predicting value of the output attribute knowing the value(s) of input attribute.

Outlook	Temperature	Humidity	Wind	Play
Sunny	85	85	FALSE	No
Sunny	80	90	TRUE	No
Overcast	83	78	FALSE	Yes
Rain	70	96	FALSE	Yes
Rain	68	80	FALSE	Yes
Rain	65	70	TRUE	No
Overcast	64	65	TRUE	Yes
Sunny	72	95	FALSE	No
Sunny	69	70	FALSE	Yes
Rain	75	80	FALSE	Yes
Sunny	75	70	TRUE	Yes
Overcast	72	90	TRUE	Yes
Overcast	81	75	FALSE	Yes
Rain	71	80	TRUE	No

Table 1: Dataset for the golf concept

Consider table 1 as an example; let us say we want to have a system that can predict if the day is worthy or unworthy to play golf. Inputs are *outlook, temperature, humidity and windy*; these are the factors that we believe they may affect the play golf concept. The output is the status as either *yes* or *no*. Therefore, *the task of supervised learning methods is to learn the mapping from input to output* (Alpaydin 2004)

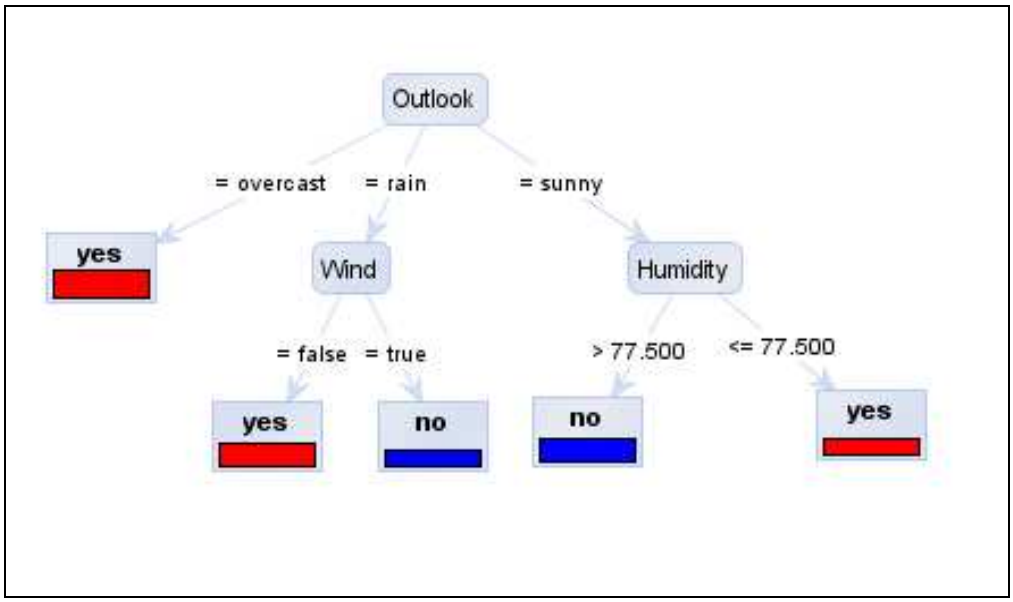

Figure 2: Decision tree for the golf concept

Supervised learning techniques provide twofold purposes. Firstly, its associated techniques are used to build classification models, decision tree in figure 1 as an example, from data sets containing examples and nonexamples of concepts to be learned. Secondly, the constructed model is used to classify newly presented instances of an unknown class (Roiger & Geatz 2002). Typical examples of techniques falling into this group include classification techniques such as decision trees and feed forward neural network and regression techniques such as logistic and linear regression (Mehrotra, Mohan & Ranka 1997).

2.3.2 Unsupervised machine learning

Unlike supervised learning, unsupervised learning builds models from data without predefined classes or examples (Caelli & Bischof 1997). This means, no "supervisor" is available and learning must rely on guidance obtained heuristically by the system examining different sample data or the environment (Mitchell 1997; Han & Kamber 2002). The output states are defined implicitly by the specific learning algorithm used and built in constraints (Caelli & Bischof 1997)

Clustering provides a concrete unsupervised learning method example (Han & Kamber 2002). Clustering is the technique used in data mining and its related activities to group or cluster observations with similar characteristics (Romesburg 2004). Instances of the data are grouped together depending on the similarity of the characteristics in the clustering system that tries to identify and extract similar groups of observation from the dataset (Raynor 1999; Han & Kamber 2002). Figure 4 represents clusters for the five musical classes; hip hop, pop, punk, electronica and netal.

The principal aim of this dissertation identified in chapter 1 is to develop the classifier workflow for the comparison of the classification techniques which falls under the supervised learning category. Therefore, from this point onwards, unsupervised learning will be considered as beyond the scope of this dissertation.

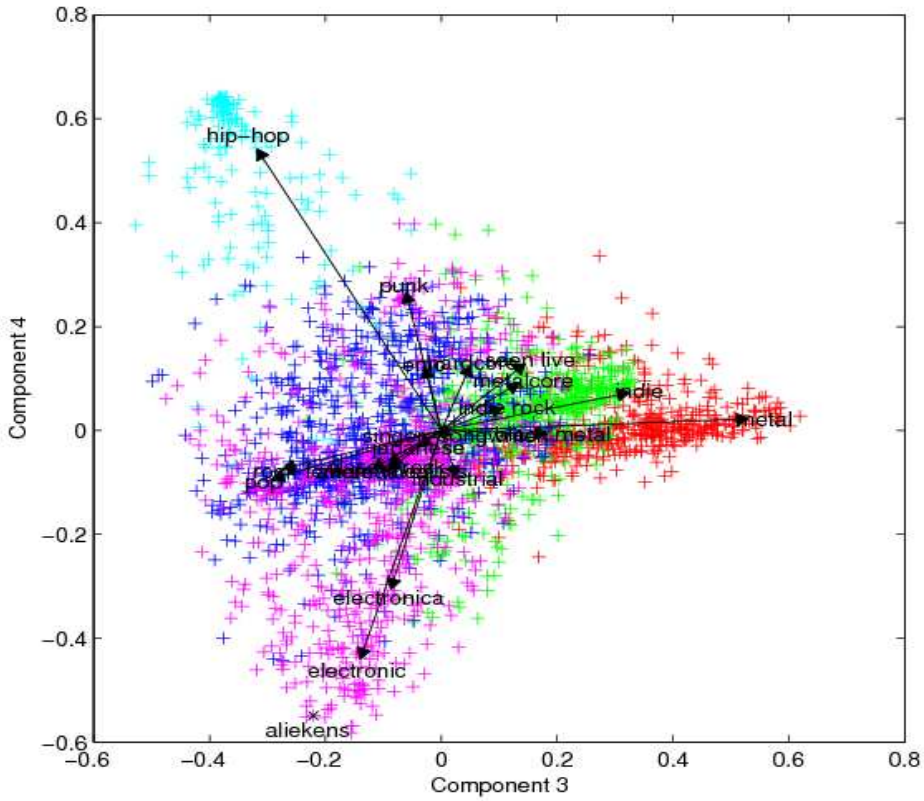

Figure 3: Clustering of data into 5 clusters

(Source: (Liekens 2007))

2.4 Classification techniques

This section provides an overview of five classification techniques. Although, there are a number of other algorithms and many variations exist depending on the application domain; only five techniques; decision tree, neural network, k-nearest neighbour, random forest and support vector machines will be discussed as this is enough to give readers' an understanding of the variations present in different approaches taken to classification. Also the presented classification techniques are the most common in modelling machine learning applications.

2.4.1 Decision trees

Decision trees are among the oldest classification technique-used for the first time in statistics and decision theory more than thirty years ago and later in other fields such as data mining, machine learning and pattern recognition (Rokach & Maimon 2005). As discussed in section 2.3.1, decision tree algorithms are categorised as supervised learning and are applicable in a wide variety of application such as credit applications.

Originally, decision trees were developed for use by statisticians in order to automate the process while determining which fields in their database were actually correlated with a particular problem they were trying to solve (Berson et al. 1999).

Cormen et al. (2003) define decision tree as "*a fully binary tree that represents the comparison between elements that are performed by a particular sorting algorithm operating as an input of a given size"*.

This definition is limited as it does not incorporate machine learning tasks that decision trees can be used. Also for those who are unfamiliar with machine learning, the definition lacks the strategy used by decision tree algorithms while creating its classifier.

A more broad definition is proposed by Alpaydin (2004) "as a *hierarchical model based on nonparametric theory where local regions are identified in a sequence of recursive splits in a smaller number of steps that implements divide-and-conquer strategy used in classification and regression tasks"*. He proposes the divide-and-conquer strategy for decision tree learning algorithms and tasks that can be performed using decision trees.

The hierarchical structure of the decision tree is divided into three parts; root node, internal nodes and leaf nodes, consider figure 1 where *outlook* is the root node, *wind* and *humidity* are internal nodes and *yes/no* are leaf nodes. The process starts at the root node and is repeated recursively until the leaf node is encountered, which provides the output of the problem. Each leaf node has an output label, which in case of classification is the classification code such as *yes/no* and in case of regression is the numeric value (Alpaydin 2004). Decision makers prefer less complex trees since the tree complexity has a crucial impact on the performance of the tree (Breiman et al. 1993; Rokach & Maimon 2005)

Decision tree learning is associated with several algorithms. The two most widely used algorithms are C4.5 (Quinlan 1993) and CART (Breiman et al. 1993) which has resulted from two different fields. C4.5 arises from the artificial intelligence

community and CART was developed in the statistics community (Craven 1996). C4.5 is the successor to the ID3 algorithm.

Given the available dataset, most common method for learning decision trees is top down induction. Start from the entire set of the training examples; partition it into subsets by testing the value of attribute and then recursively call an induction algorithm for each subset (Esmeir & Markovitch 2006). Decision trees serve several advantages to other classification techniques and to its users. For other techniques, they can be used as the first pass of a data mining run to create a subset of possibly useful predictors that can be used in other techniques such as neural networks, nearest neighbour and other statistical techniques (Berson, Smith & Thearling 1999).

For the users of the technique, decision trees have one advantage that several other techniques do not provide; interpretability. This means, a decision tree can be converted into a set of rules (*if-then*) that are easily understandable to expert and non-expert users (Berson, Smith & Thearling 1999; Alpaydin 2004). Figure 5 represent *if-then* rules extracted from decision tree provided in figure 1.With these rules users can learn and make decisions.

```
if Temperature > 84 then no  (1 / 0)
if Outlook = overcast then yes  (0 / 4)
if Humidity > 95.500 then yes  (0 / 1)
if Wind = false then yes  (1 / 3)
else no  (3 / 1)
```

Figure 4: *If-Then* rules extracted from the decision tree

Decision trees representation is considered comprehensible as they provide self explanations and when compacted becomes easy to follow (Rokach & Maimon 2005). Craven (1996) addresses a number of reasons why comprehensibility; ability to produce written communication to recipients is important in classification techniques. These are:

- **Validation:** In order to gain confidence in the performance of the learning system its users often want to know how it arrives at its decision. The ability

to inspect a learned hypothesis is important in some domains. A good example is in medical diagnosis where a system occupies a position of trust. Rules developed from the model (figure 3 and 5) can be used to validate the decisions that have been achieved by the model.

- **Discovery:** A learned model may also be important in the process of scientific discovery. A system may discover some prominent features and relationships in the training data whose importance was not previously recognised. If the hypothesis formed by the learner is comprehensible then these discoveries can be made accessible to human review. Decision trees can be used to provide such an explanation through the use of *if-then* rules if there are new important features recognised.

- **Explanation:** In some domains such as financial institutions dealing with credit applications as presented in section 2.3.2, it is not desirable to have a complete description of the learning system induced model. Rather it requires an explanation of the classification of each example. If the learned hypothesis is understandable in such domain then it can be used to produce explanation of classification of each individual case.

- **Improving predictive accuracy:** The feature representation used for a learning task can have a significant impact on how well an algorithm is able to learn and generalize. Learned models that can be learned, understood and analyzed may provide insight into devising better feature representations.

Depending on the application domain, decision trees are more preferred than other accurate techniques such as SVM due to their interpretability (Alpaydin 2004). Also decision trees have been shown that they work well experimentally (Roiger & Geatz 2002). The two most common problems of decision trees are; they suffer from overfitting; model does not perform better on unseen data and the training time in terms of speed is relatively expensive (Zhao & Sinha 2005).

2.4.2 Neural networks

Neurobiological studies are considered as the root of the artificial neural networks (ANNs) that date back about a century ago and have been seen as a motivation for the development of various algorithms (Mehrotra, Mohan & Ranka 1997). With neurology, for many decades biologists were trying to find exactly how the nervous system of the human brain works while in psychology, psychologists were trying to understand exactly how learning, forgetting, recognition and other tasks are accomplished by animals (Mehrotra, Mohan & Ranka 1997). Neural networks have been applied in a number of real world application domains ranging from pattern recognition as discussed in section 2.3.1 to medical diagnosis (C.-T. Lin & Lee 1991) as discussed in section 2.3.3. Other application domains includes forecasting and risk assessment and control systems (Rajashekaran, Pai & Vijayalksmi 2004)

The name neural networks or Artificial Neural Networks (ANN) originates from the nervous system of an animal which comprises of a large number of interconnected neurons (Mehrotra, Mohan & Ranka 1997). ANNs come in many shapes and can be used in both supervised and unsupervised machine learning problems (Mitchell 1997; Roiger & Geatz 2002). There are various neural network models exist, these are Boltzmann machine (Hinton & Sejnowski 1986), Hopfield net (Hopfield 1982), self organizing feature maps (Kohonen 2001) and the most popular one multilayer feedfoward neural networks.

Safavian & Landgrebe (1991) define "*a multilayer feedfoward neural network as an acyclic directed graph consisting of several layers of simple processing elements known as neurons"*.

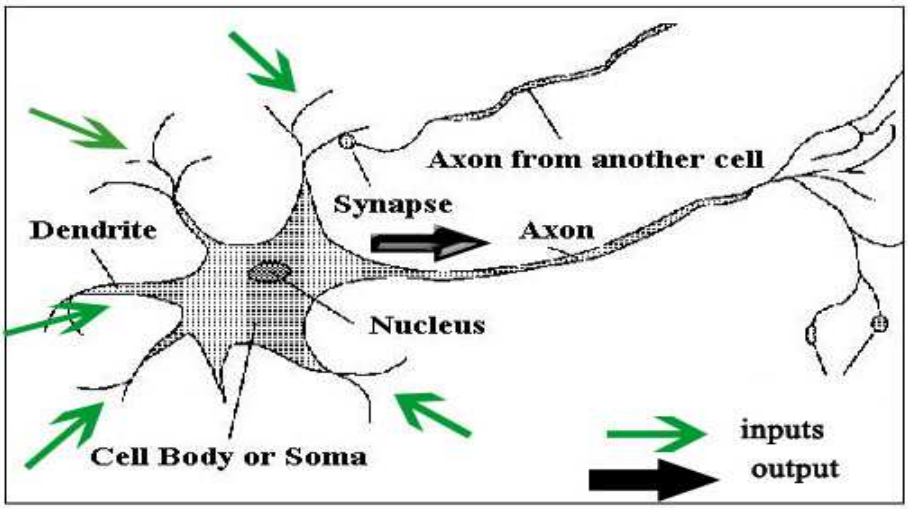

Figure 5: Biological neuron

(Adapted from: (Gurney 1997))

ANN is made up of two parts as related to human brain shown in figure 5, namely, a *node* which loosely corresponds to a neuron and a *link/connection* which corresponds to connections between neurons (Craven 1996). In a neural network, each node performs some simple computations and each connection conveys a signal from one node to another labelled by numbers called *weights or connection strength* (Mehrotra, Mohan & Ranka 1997). ANNs are made up with three layers, namely input, hidden and output layer(s). The input values, such as, $x_1, x_2, x_3, x_4 \cdots x_n$ in figure 4, to the network are supposed to be numeric (Roiger & Geatz 2002).

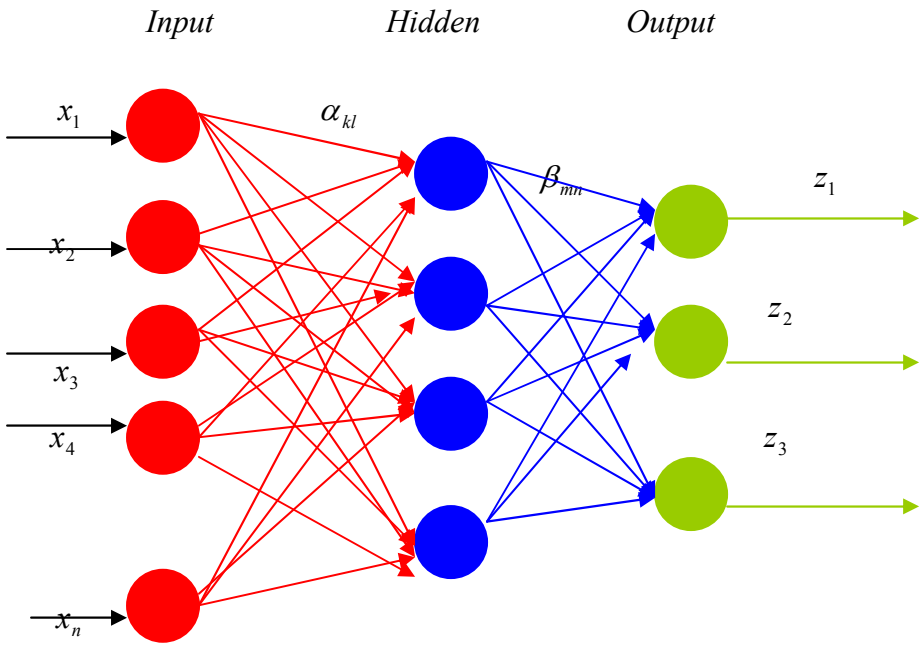

Figure 6: A three layer feed forward neural network with 5 inputs and 3 outputs. (Source: (Safavian & Landgrebe 1991))

From figure 6, the feed forward neural network is made up of three layers; input, hidden and output. $x_1, x_2, x_3 \cdots x_n$ are the input values presented in the input layer while z_1, z_2 *and* z_3 are output values. α_{kl} and β_{mn} are the *connection weights* between neuron i of input layer to neuron j of hidden layer and neuron m of hidden layer to neuron n of output layer. The output values are formed as the product of the input value and its related connection weight.

Neural networks have several advantages compared to other techniques, among the advantages are; they provide highly accurate predictive models in complex domains and provides fast testing speed as they are considered as the eager learners (Kostiantis 2007). Eager learners generalize before seeing query while lazy learners wait for a query before generalizing.

Despite having the aforementioned advantages, the algorithm is associated with several disadvantages. Complexity in use and limitation in deployment (Berson, Smith & Thearling 1999) are among of its weaknesses and the training speed is relatively slow (Berson, Smith & Thearling 1999). When comparing to the previous

technique; decision tree which provides explanation; neural networks are difficult for the users to interpret (Campbell 2000)

Neural networks have been successfully used in a wide variety of practical classification problems, such as recognising printed or handwritten characters, classifying loan application into credit-worthy or non-credit-worthy and analysing sonar and radar data to determine the source of the signals (Mehrotra, Mohan & Ranka 1997).

2.4.3 K-nearest neighbour

K-Nearest Neighbour (KNN) is of the methods referred to as instance-based learning which falls under the supervised learning category (Mitchell 1997). This technique is quite different from all of the other techniques discussed so far. KNN works by simply storing the presented training data (Mitchell 1997). When a new query or instance is fired, a set of similar related instances or neighbours is retrieved from memory and used to classify the new instance (Mitchell 1997; Han & Kamber 2002). While classifying, it is often useful to take more than one neighbour into account and hence refereed to as *k-nearest neighbour* (Cunningham & Delany 2007). KNN has been applied in a number of application domains ranging from webpage categorisation to credit risk assessment.

KNN is considered as the most basic instance-based method. This algorithm assumes that instances correspond to points in *n* dimensional space (Mitchell 1997). The classification of the instances is quite straight forward as examples are classified based on the class of their nearest neighbours (Cunningham & Delany 2007). The nearest neighbours to an instance are measured in terms of the Euclidean distance and some other related measures. Euclidean distance measures the dissimilarities between examples represented as vector inputs.

Considering the instance y be described as

$$a_1(y), a_2(y), a_3(y).....a_n(y)$$

Where $a_r(y)$ denotes the value of r[th] attribute of the instance *a*. The distance d between two y_i and y_j points can be calculated by Euclidian distance as follows:

$$d(y_i, y_j) = \sqrt{\sum (a_r(y_i) - a_r(y_j))^2}$$

The basis for classifying a new query using Euclidean distance is that, the classification of an instance will be most similar to the classification of other instances that are in the same group. In a more simple way this can be concluded as, instances in the same group are expected to have a small separating distance compared to instances that fall under different groups. KNN like other classification techniques has several advantages and disadvantages depending on the amount of training data provided.

KNN is easy to understand and simple to implement due to its transparency in processing (Cunningham & Delany 2007). This technique should be considered when seeking solution to any classification problem as there are some noise reduction techniques that work only for this technique and can be used to improve the performance of the classifiers (Cunningham & Delany 2007). In some application domains, explanation of the output of the classifier*, interpretability*, is of huge importance, if this is the case then KNN can be very effective if the analysis of the neighbours is as useful as explanation. In case of the training speed, KNN is among the techniques that provide faster training speed.

However, the algorithm is associated with high computational cost. Every time when a new query is fired and KNN algorithm needs to classify the new instances, related instances are retrieved from memory and this result in to low testing speed (Zhou, Ooi & Meng 2005). Also instance based methods in particular KNN, tend to consider all attributes of the instances when attempting to retrieve similar training examples already stored, curse of dimensionality while in decision tree learning systems, only relevant attributes are forming the hypothesis.

Figure 7 is an example of the application of the k-nearest neighbour. There are three types of classes; *circles*, *triangles* and *rectangles* and newly fired *cross* instance

which is an unknown instance that needs to be classified. Unknown class is near by all three known classes; circles, triangles and rectangles. Distance between classes will help in classifying cross instance as to which group it belongs.

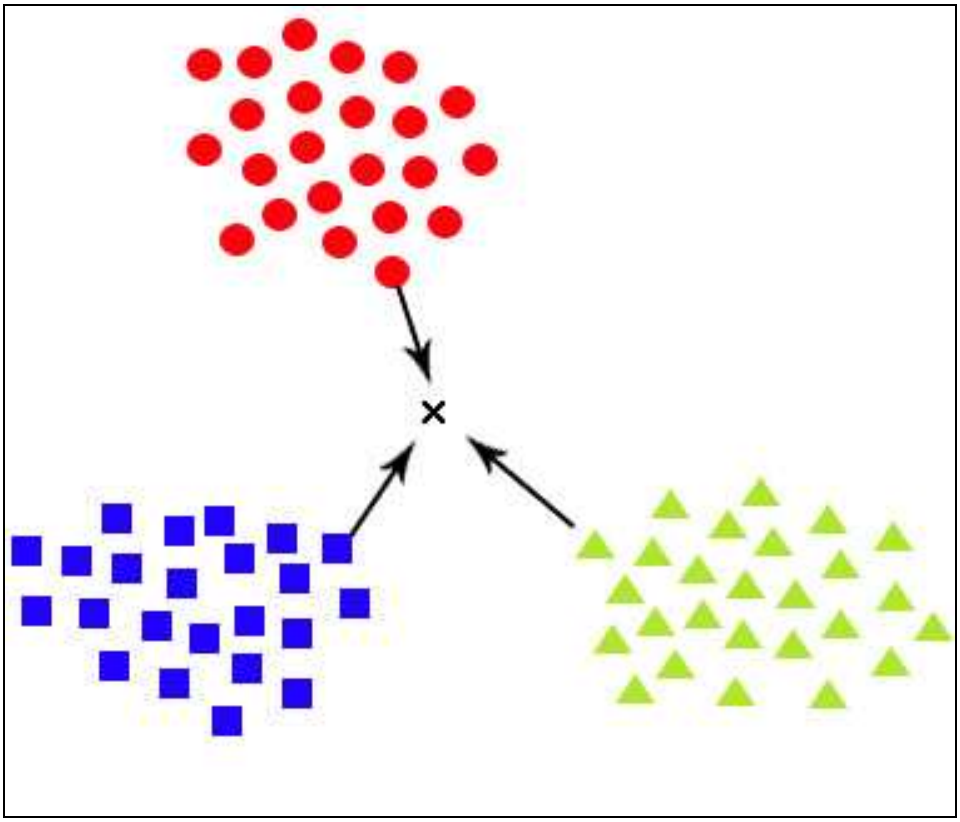

Figure 7: KNN classifier for three classes and a new instance

2.4.4 Support Vector Machine

In recent years, a new community involving several researchers and engineers has emerged with useful text books, web sites and conferences in a new machine learning technique. The focus of this research is based on Support Vector Machines (SVM) and other Kernel methods such as regression, density estimation and kernel PCA (Chih-Jen Lin 2006). The growing interest of many researchers in this area is driven by several advantages that these algorithms provide compared to other older algorithms (David J. Hand 2006).

The SVM is a relatively new machine learning technique proposed by Vladimir Vapnik and his team at AT&T Bell laboratories in 1992 which utilizes techniques from statistics, optimization and functional analysis (Osuna, R. Freund & Girosit

1997). It represents the state of the art in machine learning techniques and has managed to achieve competitive results in both classification and regression tasks (R. Stolean et al. 2007). It has since been, studied, greatly generalized and applied to a number of real world applications in different fields such as text categorization, hand written character recognition, image classification and medical and biological information analysis (Tang et al. 2004; Winkler, Niranjan & Lawrence 2005).

The general idea of the SVM is to find separating hyperplanes between training instances that maximize the margin and minimize the classification errors (Campbell 2000). Margin or sometimes refereed to as geometric margin is refereed *as the "distance between the hyperplane separating the two classes and the closest datapoints to the hyperplane"* (Berthold & Hand 2003). SVM and other kernel methods work with linearly and nonlinearly separable problems in classification and regression tasks. For this thesis, only classification tasks will be considered.

For the two classes' problem, there exists a dataset D with labelled examples. Having the training data which is known to be linearly separable, there exist a linear hyperplane that performs the partition of these two classes. Consider equation 1 where y_i denotes the label of instance i for the classification problem. If the label of the instance is 1, then it will be classified as an example in class 1 otherwise if the label is -1, an example will be classified as in class 2.

$$y_i = \begin{cases} 1 \text{ if } x_i \text{ in class 1} \\ -1 \text{ if } x_i \text{ in class 2} \end{cases} \qquad \text{(Chih-Jen Lin 2006)}$$

Support vector machine can separate these two classes by considering the maximum margin between the two classes which also provide small classification error. Figure 8 is an example of two classes'; rectangles and circles' which are linearly separable. Figure 8(a) has the separating hyperplane with small margin and figure 8(b) has the separating hyperplane with large margin. A better generalization capability is expected to be provided from figure 8(b) as most of the rectangles are in the upper part and most of the circles are located in the lower part. x_1, x_2, x_3 are examples of support vectors for the two classes which are classified.

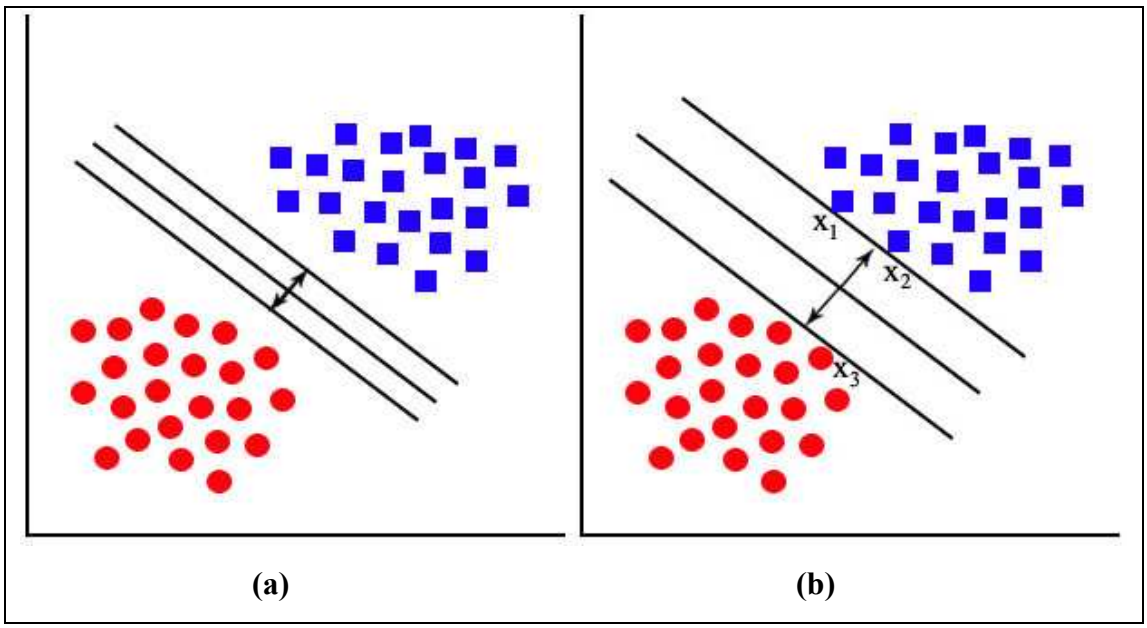

Figure 8 (a) A Separating Hyperplane with small margin. (b)A Separating Hyperplane with large margin. A better generalization capability is expected from (b). (Source: (Osuna, R. Freund & Girosit 1997)

With non-linearly separable classification problems, using linear hyperplane will not yield the best partition without any errors (R. Stolean et al. 2007). To work with non-linearly separable problem, SVM has to be defined with another characteristic; "*error bound does not depend on the dimension of space"* (Campbell 2000). This feature enables the experimenter to give an alternative kernel representation of the data which equals to mapping the data into higher dimensional space *"kernel trick"* where the two classes are more linearly separable (Bernhard 2002). After mapping the data into higher dimensional space, then linear classifier can be applied.

Figure 9 on the left hand side is an example of nonlinearly separable classification problem having two classes; circles and cross signs. In any possible way, these two classes can not be linearly separable without mapping them into a higher dimensional space. On the right hand side is a 3-dimensional space for the problem transformed from the figure in the left hand side.

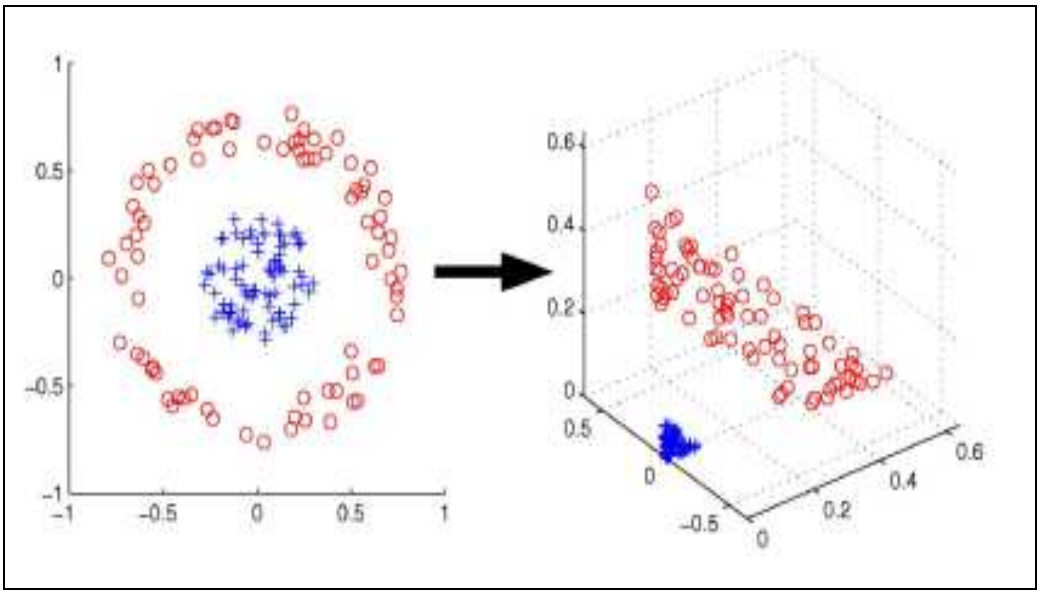

Figure 9: On the left hand side, non-linearly separable problem containing circles and positive signs. Right hand side, a linearly separable problem mapped into 3D space (Source: (Holbrey n.d))

Moreover, advantages of SVM and other kernel methods are worth mentioning. Firstly, they are explicitly based on the theoretical model rather than on loose analogies of the natural learning systems. Secondly, Kernel methods are free from the problems of local minima as in neural network, (Campbell 2000). Also they came with theoretical guarantees about their performance (Chih-Jen Lin 2006) and have modular design that makes it possible to separately implement and analyse their components (Campbell 2000).

Successful application in SVM ranges from text categorization, handwriting recognition and medical, pattern recognition and biological information analysis (Campbell 2000).

2.4.5 Random forests

Random forest involves the generation of an ensemble of trees that vote for the most popular class (Breiman 2001). With respect to other classification techniques discussed so far, random forests have two distinguishing characteristics; the generalization error converges as the number of trees in the forest increases and the technique does not suffer from overfitting (Breiman 2001). Accuracy of the individual

single trees that make up a forest enforces the convergence of the generalization errors and hence improvement in classification accuracy.

Breiman (2001) defines a random forest as a *classifier consisting of a collection of tree-structured classifiers {h(**x**, $Q_{k,}$ k=1…} where the {Q_k} are independent identically distributed random vectors and each tree casts a unit vote for most popular class at input **x**.*

Breiman (Breiman 2001) proposes that *"in order to grow an ensemble of trees, often random vectors are generated that govern the growth of each tree in the forest"* (Breiman 2001). Several examples of random vectors exist such as bagging or bootstrap aggregating (Breiman 1996) which is regarded as the most straightforward way of manipulating training data. With bagging, given a training set *S* of *m* examples drawn at random, a new training set S' is constructed by drawing *m* examples uniformly with replacement from *S* (Dietterich 2000a). Other examples of random vectors are Adaboost algorithm or boosting (Freund & Schapire 1996) and random spit selection (Dietterich 2000b).

The Adaboost algorithm, developed by Freund & Schapire (1996) manipulates training examples to generate multiple hypotheses. It maintains a set of weights over the original training set and adjusts these weights after each classifier is learned by the base learning algorithm (Freund & Schapire 1996). While aiming at minimizing the weighted error of the training set, in each iteration *l*, the learning algorithm is invoked and it returns hypothesis h_l. The weighted error of h_l is computed and applied to update the weights on the training examples (Dietterich 2000b).

Dietterich (2000b) introduces random split selection which is a modified version of the C4.5 (Release 1) learning algorithm in which the decision about which split to introduce at each internal node of the tree is randomized. Bagging tends to work well with unstable learning algorithms (Dietterich 2000b)–these are algorithms whose prediction undergoes large changes in response to small changes in training data. Examples of unstable learning algorithms are neural networks, decision trees, regression trees and rule learning algorithms (Dietterich 2000b). The linear threshold

algorithm, linear regression and nearest neighbour are examples of stable learning algorithms.

Several experimental studies (Bauer & R. Kohavi 1999; Dietterich 2000a) have been done while comparing the accurateness of these ensemble methods. Adaboost has shown to provide better results compared to bagging and random split selection (Xiao-Dong Liu, Chun-Yi Shi & Xue-Dao Gu 2005). Bagging and randomized trees provide similar performance; the difference comes when randomization can do better in some cases while bagging on every large dataset (Dietterich 2000a)

For the random forests, Breiman (2001) asserts several characteristics for its accuracy. Firstly, random forests converge. Given an ensemble of classifiers $h_1(x), h_2(x), h_3(x) \cdots h_k(x)$ and with the training set drawn at random from the distribution of random vector Y, X defines the margin function (mg) as

$$mg(X,Y) = av_k I(h_k(X) = Y) - \max_{j \neq Y} av_k I(h_k(X) = j)$$

The margin; *distance between hyperplane and the nearest point*; measures the extent to which the average number of votes at **X**, Y exceeds the average vote for any other class. This comes into conclusion that, the larger the margin, the more confidence in the classification.

The second characteristic of the random forests is, for the generalization error of the technique an upper bound can be derived from two parameters; measures of how accurate the individual classifiers are and the dependence between the classifiers (Breiman 2001). This results into the classifier not to suffer from overfitting.

For two class problems, Breiman (2000) shows that random forest is equivalent to kernel running on its true margin. The argument provided is, "*randomness provides the symmetry of the kernel while strength enhances a desirable skewness at abrupt curved strength"*.

2.5 Existing workflows

For any application domain that needs a classifier to be developed for the sake of solving problems that arise, machine learning and data mining offers a number of workflows which provide guidance for the experiments in related projects. These workflows have been developed by industries and machine learning and data mining researchers and aims at providing procedural steps and a list of tasks that are supposed to be performed by the experimenter while performing experiments.

For this dissertation only three workflows will be discussed which provides guidance while performing machine learning and data mining projects. These are Cross Industry Standard Process for Data Mining abbreviated as CRISP-DM (http://www.crisp-dm.org), KDD process model and its variations (Fayyad, Piatetsky-Shapiro & Smyth 1996; Collier et al. 1998; Feldens et al. 1998) and SAS-SEMMA from the SAS Company (http://www.sas.com). For the classification research while comparing performance of the classification techniques, these workflows provide guidance on a number of steps.

This section provides an overview of these three common workflows used in machine learning and data mining projects. The workflows and their structures are explored and the data or information flow is identified. The author discusses and compares the workflows as thorough as possible with each other. The author also indicates the advantages and disadvantages of using certain steps in existing workflows for small scale classification projects. Both of these advantages and disadvantages will add to the creation of a more comprehensive and suitable classifier workflow proposed in chapter 5.

2.5.1 CRISP-DM

CRISP-DM stands for CRoss Industry Standard Process for Data Mining developed in 1997 by two vendors ISL (now part of SPSS (http:// www.spss.com)) developers of the market-leading Clementine Data Mining System and NCR Corporation; the world's leading supplier of data warehouse solutions along with two industrial partners; Daimler-Benz (now DaimlerChrysler (http:// www.daimler.com)) and

OHRA, one of the largest Dutch insurance companies (by the year 1997) . It is a general purpose process model for carrying out projects in varying applications.

The methodology applied to the CRISP-DM lifecycle is of the hierarchical origin (from phases to process instances) as shown in figure 10. This methodology can be broken down into four levels, namely; phases, generic tasks, specialised tasks and process instances. For the methodology in relation to data mining projects, developers of the CRISP-DM assert that, there possibly exists a relationship between all data mining tasks caused by goals, background and interest of the user.

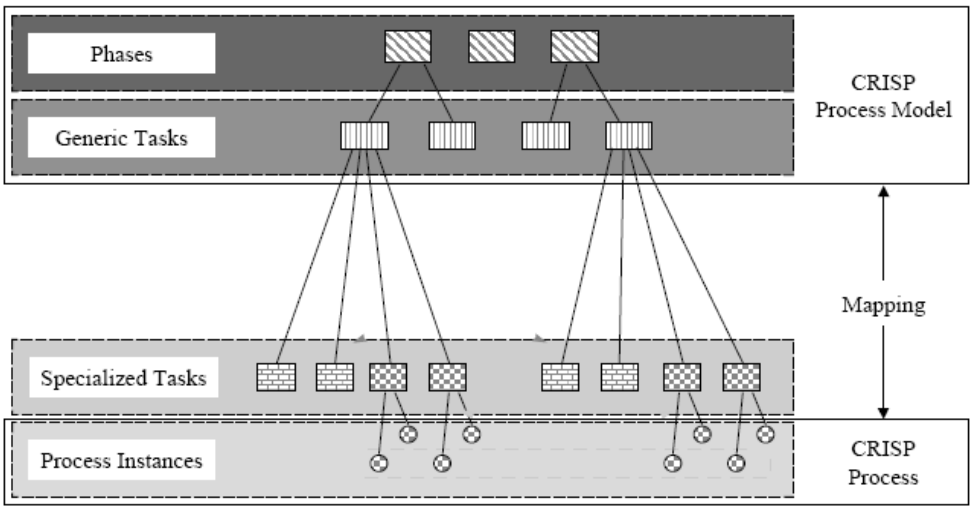

Figure 10: CRISP-DM methodology (Source: (CRISP-DM 2000))

The aim of CRISP-DM is to make large data mining projects less costly, more reliable, more manageable, more repeatable and faster (Wirth & Hipp 2000). This process model is independent of both industry sector and technology used, as it can be integrated with any industry standard process such as SAS-SEMMA using any technology (Wirth & Hipp 2000).

CRISP-DM Phases

For large projects, the CRISP-DM process model is useful for planning, communication within and outside the project team and documentation (Wirth & Hipp 2000). The process is made up of six phases, namely business understanding, data understanding, data preparation, modelling, evaluation and deployment that make

up the lifecycle of the project. Some phases go back and forth, as shown in figure 11, depending on tasks that are processed in such a phase.

- **Business Understanding:** This is the initial phase of the project and aims at understanding project objectives and requirements from the business perspective, and transforming acquired knowledge into a data mining problem definition. This is where a preliminary plan is designed for the purpose of achieving the objectives of the project.

- **Data Understanding:** This phase deals with all the activities related to data manipulation. It starts with the data collection and proceeds with other data manipulation activities in order to get familiar with the data. The data understanding phase aims at identifying data quality problems or detecting interesting subsets of the data to form hypothesis for hidden information.

- **Data Preparation:** After understanding the raw data the data preparation phase deals with all of the activities required to construct the final dataset. It is a process which is performed multiple times without prescribed order. After this phase, collected data are fed into the modelling tool(s). This phase also involves tasks such as feature and record selection and cleaning, and transformation of the data.

- **Modelling:** Within the modelling phase, modelling techniques are selected and applied, and their values are calibrated to optimal values. Different techniques require different forms of data. This is why there is a going back and forth between data preparation and modelling. The outputs of this phase are model(s) which need some evaluation. Models developed can be used to increase knowledge of the data or the knowledge gained will need to be organised and presented in a way that customer can use.

- **Evaluation:** At this phase of the project, models with high quality have already been built in the modelling phase using data extracted from the data

preparation phase. Built models are evaluated and the steps followed when constructing the models are reviewed. The evaluation is done in order to be certain that developed model(s) achieve business objectives set in the first phase.

- **Deployment:** This is the last phase of the CRISP-DM process model. This phase becomes effective if the model developed in the fourth phase needs to be organised and presented in a way that a customer can use. As deployment is not the effort of the analyst, the customer needs to understand a set of actions that need to be taken in order to make use of the created model(s). Figure 11 shows the six phases of the CRISP-DM process model.

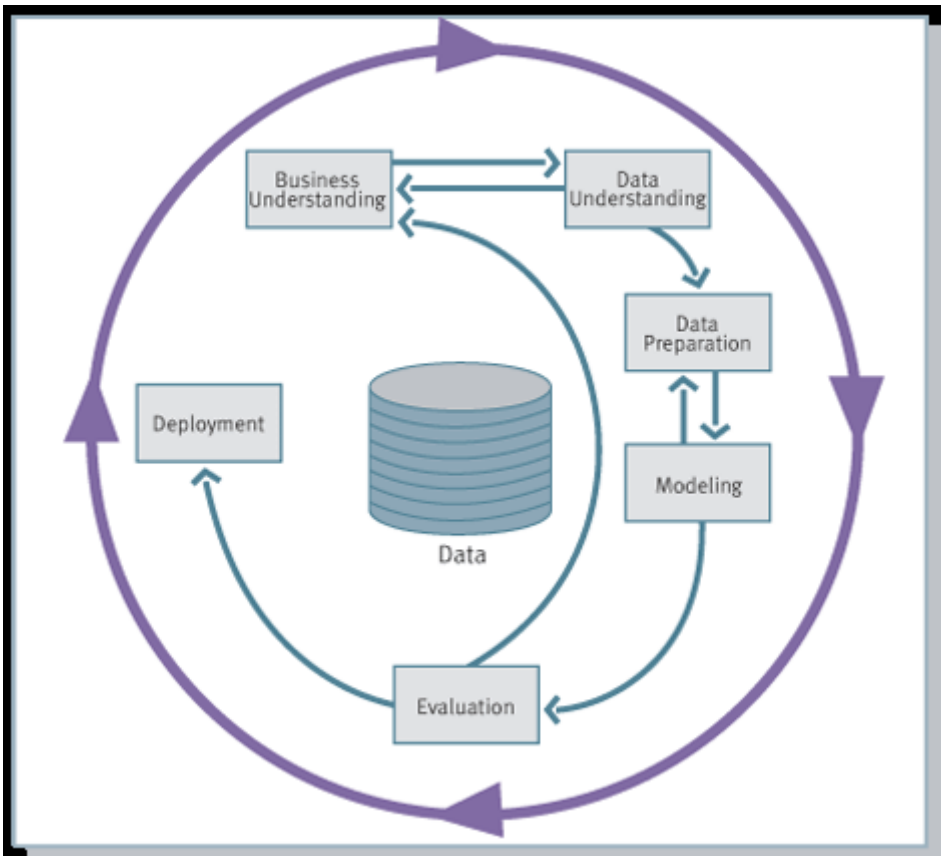

Figure 11: CRISP Data Mining process model
(Source: (CRISP-DM 2000))

2.5.2 KDD Process Model

This section provides the discussion about the KDD process model developed by Fayyad et al. (1996) and its variations developed by (Collier et al. 1998) and (Feldens et al. (1998). The KDD process model was developed to represent a set of processes for discovering useful knowledge from data (Fayyad, Piatetsky-Shapiro & Smyth 1996; Collier et al. 1998). This process model has evolved and continues to evolve in various fields such as machine learning, statistics and pattern recognition, artificial intelligence and reasoning with uncertainty, and information retrieval. In this dissertation, the discussion about the KDD process has been divided into three parts. The first part introduces the traditional KDD process model developed by Fayyad et al. followed by its variations; the iterative KDD process model and the integrated KDD process model

2.5.2.1 Traditional KDD process model

Despite having a huge amount of authors (Collier et al., Feldens et al.) commenting on the original KDD process model, Fayyad et al. are recognised as the core authors of the process model.

Fayyad, Piatetsky-Shapiro & Smyth (1996) define the KDD as the *"nontrivial process of identifying valid, novel, potentially useful and ultimately understandable patterns in data"*.

Traditional KDD process steps

Fayyad et al. (1996) identifies the KDD process model made up of nine steps as interactive and iterative as much of the decisions are made by the user. As shown in figure 12, the steps involved are learning the application domain, creating the target dataset, data cleaning and preprocessing, data reduction and projection, choosing the function of data mining. Other steps are choosing the data mining algorithm, data mining, interpretation and evaluation and using the discovered knowledge.

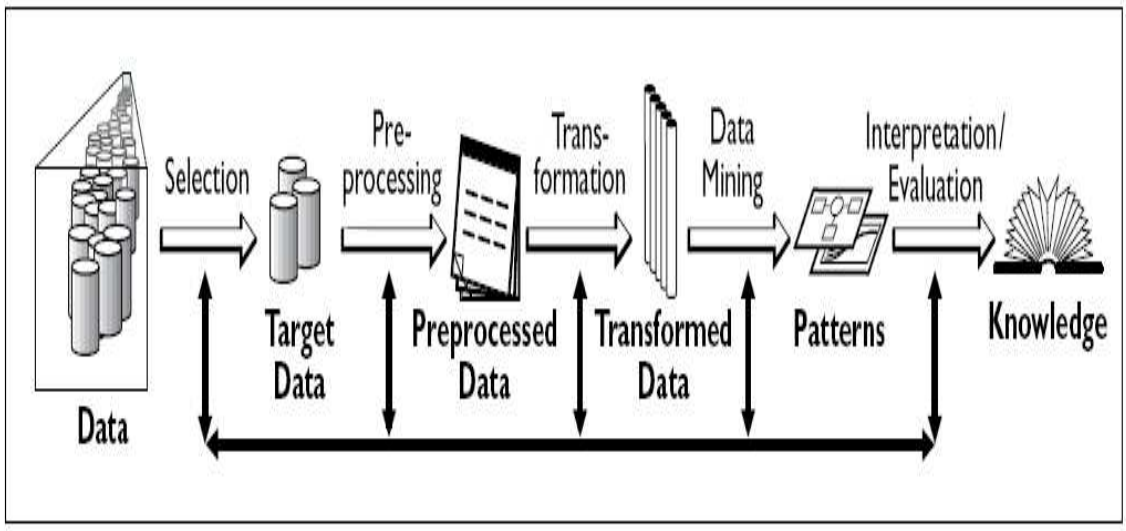

Figure 12: Overview of the steps constituting KDD process
(Source: (Fayyad, Piatetsky-Shapiro & Smyth 1996))

- **Learning the application domain:** This step involves establishing the goal of the application or setting the objectives of the experiment. The experimenter is supposed to understand the relevant prior knowledge from the data and the objectives of taking the experiment.

- **Creating a target dataset:** As most of the datasets produced nowadays weigh many gigabytes, the second step in the KDD process model is creating the target dataset. This is done by selecting the representative dataset or selecting the subset of variables or data samples on which the experiments will be performed. The outcome of this step is the target dataset as shown in figure 12.

- **Data cleaning and preprocessing:** This is the most important step in machine learning projects as the processes performed in this step can change the overall results if not well performed. It constitutes collecting necessary information to model, deciding on the strategies for handling missing data fields and removing noise or outliers if necessary. The outcome is the preprocessed data where data problems have been dealt with.

- **Data reduction and projection:** Data are reduced in such a way as useful features to represent the data are selected depending on the goal of the tasks that has been set in the first step. Also transformation methods are employed

to reduce the effective number of variables under consideration or to find invariant representation of the data.

- **Choosing the function of data mining:** This step involves deciding the purpose of the model derived by the data mining algorithm. Fayyad, Piatetsky-Shapiro & Smyth (1996) asserts a number of purposes that can be selected by the experimenter, namely summarisation, classification, regression and clustering.

- **Choosing data mining algorithm:** Includes selecting method(s) to be used while searching for the patterns in data and deciding which models and parameters may be appropriate. Different algorithms exist ranging from decision trees, neural networks to support vector machines as discussed in section 2.5.

- **Data Mining:** This step involves searching for the patterns of interest in a particular representational form or a set of such representations including classification rules, clustering, sequence modelling, and dependency and line analysis. The results of the data mining phase are the patterns.

- **Interpretation/ Evaluation:** It involves interpreting the discovered patterns and possibly returning to any of the previous steps as well as possible visualisation of the discovered patterns. Removing redundant or irrelevant patterns and translating the useful ones into terms understandable by the users. The result of this step is the discovered knowledge.

- **Using discovered knowledge:** Sometimes the discovered knowledge is needed for the customers' systems. The discovered knowledge after interpretation is incorporated into the performance system or simply documenting it and reporting it to interested parties as well as checking potential conflicts with previously extracted knowledge.

From the discussed phases, the KDD process model lacks the deployment step which is necessary if the discovered knowledge needs to be transformed to the users' system.

This resulted into a number of variations (Collier et al. 1998; Feldens et al. 1998) from the traditional KDD process model. These variations raise one or more arguments commenting why their authors (Collier et al. 1998; Feldens et al. 1998) think the traditional KDD process model is not complete. CRISP-DM (2000) identified this issue and modified their workflow by incorporating the deployment phase.

2.5.2.2 Iterative KDD process model

Collier et al. (1998) modified the traditional KDD process model by Fayyad et al. and introduced an iterative KDD process model. The modification of the traditional KDD process model was enforced by three questions posed by Collier et al. which shows the traditional process was incomplete.

Comparing the first step of the traditional KDD process model and iterative KDD process model provided in figure 12 and 13 respectively; there is a change from learning the application domain to define objectives. Collier et al. claim that there is a common misconceptions about data mining as one can set algorithms loose on the data to find all interesting patterns without understanding the business needs and relating them to the objectives of the experiments. They proposed the first step of the KDD process model to be determination of the objectives or goals.

The traditional KDD process model starts with the learning application domain step and ends with the interpretation. The process does not state after acquiring the knowledge from data, what next have to be done. Collier et al. (1998) introduces the deployment step where the discovered knowledge or results are deployed or re-iterated. This question has been named as actionable results and makes the final step to the iterative KDD process model.

The last contribution made by Collier et al. is related to iteration. With the traditional KDD process model, steps are one way; there is no going back to the previous step(s). Iterative KDD process model allows the experimenters to return to the previous phases to improve performance of the algorithms and in general to improve data mining results.

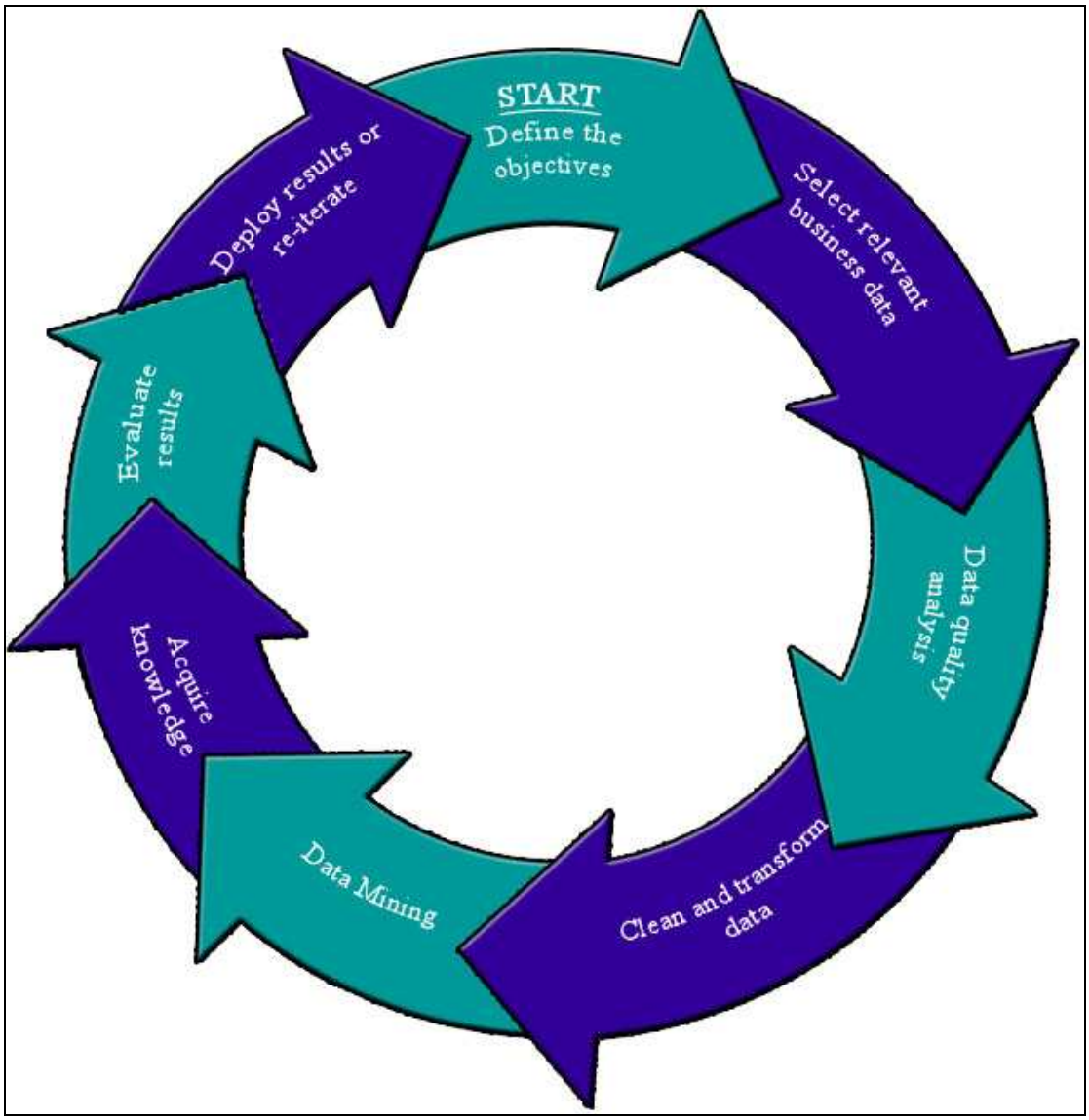

Figure 13: A refined KDD process
(Source: (Collier et al. 1998))

2.5.2.3 Integrated KDD Process

Feldens et al. (1998) studied the traditional KDD process model and provide another variation refereed to as integrated KDD process model. According to the integrated KDD process model, data warehousing methodologies and visualisation techniques play an important role in successful KDD. Integrated KDD was characterised as strongly application-oriented, iterative, interactive and non-linear (Feldens et al. 1998). Pre-processing, data mining and post-processing have been identified as the core processes that make-up the integrated KDD process as shown in figure 14.

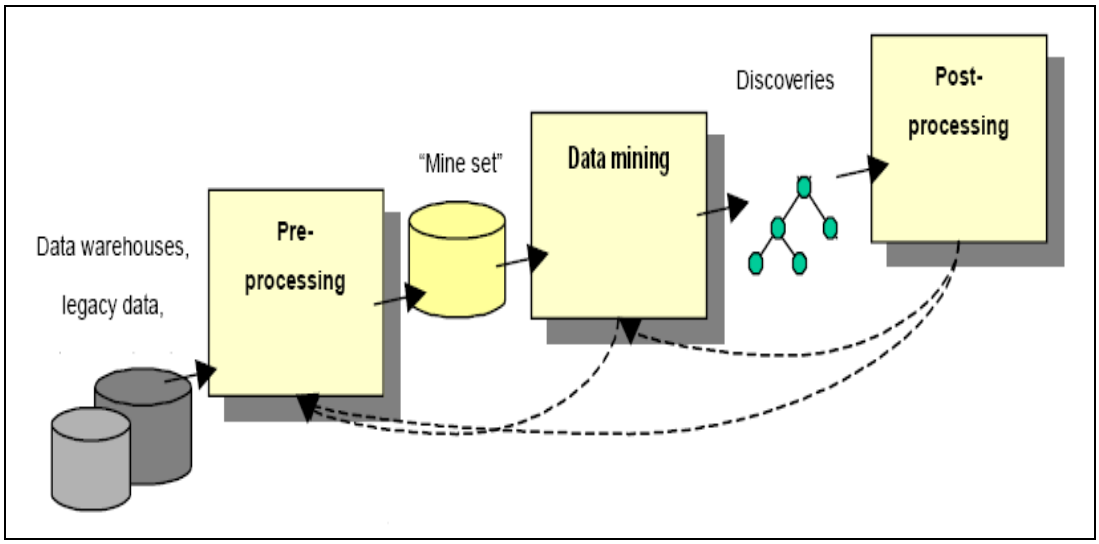

Figure 14: KDD process
(Source: (Feldens et al. 1998))

The pre-processing step includes everything that is done before data mining. Several tasks are performed, these are; analysis of the existing data, integration of the data sources and data transformations (Feldens et al. 1998). Data warehouses and legacy data are fed into the pre-processing step.

Data mining steps constitute applications of such algorithms possibly the repeated application and tuning the learning algorithm parameters to get the best learning performance. Algorithms to be used in data mining can also be chosen based on the analysis that is done to support preprocessing steps.

Post-processing is the last step where filtering of potentially useful and interesting knowledge after data mining is performed. Other tasks apart from filtering include structuring and sorting and then knowledge is presented to the user. Figure 14 represent an integrated KDD process model.

2.5.3 SAS-SEMMA

The SEMMA acronym stands for Sample, Explore, Modify, Model and Assess that represent the core processes for conducting data mining project using SAS Enterprise Miner. Having a statistical representative of the data, SEMMA makes it easy to apply exploratory statistical and visualisation techniques, select and transform most significant variables, model variables to predict outcome and confirm model accuracy.

This section provides five processes, namely sampling, exploring, modifying, modelling and assessment performed using SAS Enterprise Miner in data mining projects. The good thing about this methodology is that it can be integrated with other process models such as CRISP-DM. Despite providing the integration capability, the methodology is industry specific; it is not applicable if the experimenter is not using SAS Enterprise Miner.

SAS-SEMMA phases

The SAS-SEMMA focuses on model development aspect of data mining. The process starts with the whole datasets in its first step followed by several other steps until the final model is developed. This section provides an overview of the steps involved in SAS-SEMMA.

- **Sampling:** This is the initial stage which works by extraction of a portion of large dataset known as representative dataset. The representative dataset extracted from large dataset needs to be big enough to contain enough information and yet small enough to be manipulated quickly. The data mining process is practised in a representative data instead of the whole volume as this reduces preprocessing time required to get crucial business information. The representative data is partitioned into training, testing and validation (SAS Institute 2003).

- **Exploration:** This works by searching unanticipated trends and anomalies in order to gain understanding and ideas from the data. Two different types of visualisation can be used during the exploration step. There is visual exploration and statistical techniques. If visual exploration can not reveal clear trends then statistical techniques can be used. One of the statistical techniques that can be used is clustering as discussed in section 2.3.2. (SAS Institute 2003) asserts that an experimenter using SAS EM in explore phase is supposed to plot the data, obtain descriptive statistics, identify important variables and perform association analysis.

- **Modification:** This is done by creating, selecting and transforming the variables to focus the model selection process. Sometimes this step is known as data manipulation. One main purpose of data manipulation is to select subsets of attributes of interest in order to reduce number of predictors to be used in modelling. The whole process of selecting variables for modelling is called feature selection. Feature selection is important as it can simplify data description and in turn makes for easier understanding of the problem. Also outliers are checked; outliers are *"instances that do not obey the rule and are exceptions"* (Alpaydin 2004).

- **Modelling:** Given data are modelled by allowing the software, SAS Enterprise Miner, to search automatically for a combination of data that will reliably predict a desired outcome. Several modelling techniques exist such as decision trees, neural networks and other statistical models which comprise of time series analysis, principal component and memory based reasoning. Each model has its strengths and weaknesses depending on data provided.

- **Assessment:** This is the final step where by competing developed models in the fourth step are compared by focusing on usefulness and reliability of the findings from the data mining process and estimate how well the model performs. Models are assessed depending on performance measure(s); accuracy, error rate, precision, recall and ROC curve. In SAS Enterprise Miner the comparison is done by the model comparison icon as shown in figure 4 where decision tree, neural network and regression techniques are compared.

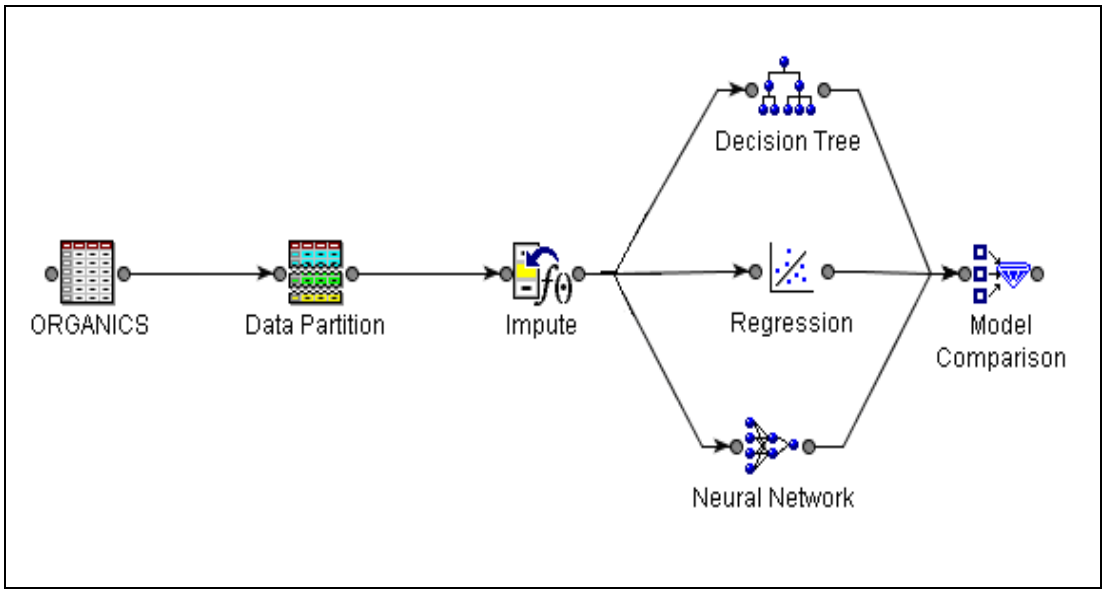

Figure 15: Layout of the SAS Enterprise Miner workflow for the comparison of decision tree, regression and neural network models

Figure 16 represents a flowchart for the SEMMA design. It consists of five SEMMA steps together with tasks that can be performed at each step. Sampling is an optional step that's why there is yes/no option. For exploration, data visualisation or statistical techniques such as clustering can be performed. In modification variable selection and data transformation are the tasks to be performed. Fourth step is modelling where models such as neural network, decision trees can be developed. The final step is assessment where developed models in step four are assessed.

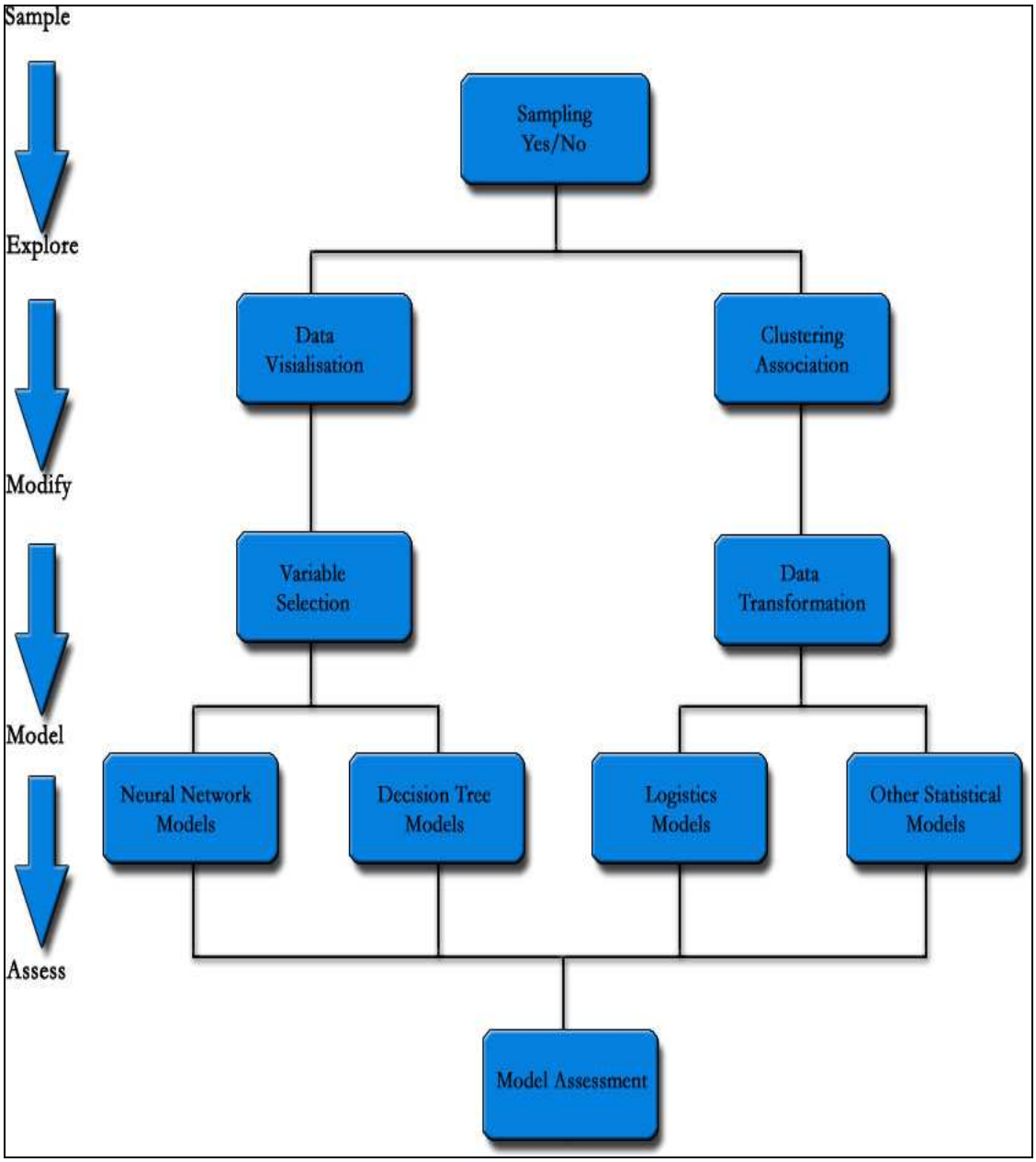

Figure 16: Flowchart to the SEMMA Design
(Source: (Matignon, Institute & I. NetLibrary 2007))

These three workflows, CRISP-DM, KDD and SAS-SEMMA are mostly used for large machine learning and data mining projects. They tend to work well with large projects as team members can divide themselves depending on the tasks and subtasks that are involved in such a project in order to achieve a desired goal. Even in large industries the workflows provides better ways for the experimenters' to perform the required tasks in a descriptive manner.

2.6 Existing workflows evaluation

From the research literature discussed in section 2.6, it is clear that the existing workflows CRISP-DM, the KDD process model and SAS-SEMMA together with their respective phases and steps are useful for large machine learning and data mining projects. However these workflows do not provide clear procedural steps and tasks that an experimenter has to follow while creating models in a single application domain where the project may not be of the same breadth as those for which these bigger workflows were designed. With this lack of clear procedural steps and tasks for small scale projects, there is a need for a workflow which can be applied to small scale classification projects.

For better performance of the classifiers from machine learning and data mining projects, any existing workflow applied in such a project has to be used by the expert user(s). This has been identified as the challenge in the development of the classifier workflow. A good model or framework has to allow all types of users; experts and non-experts. Taking the existing workflows such as SAS-SEMMA as the case study using its phases, it is difficult for non-expert users without experience in machine learning and data mining to follow the five phases without guidance from an expert. There is a big chance of generating invalid conclusions when non-expert users perform experiments using large scale workflow into a small scale project. To minimize this chance the classifier workflow will be developed to be used even with non-expert users to perform experiments for small scale projects.

The classifier workflow will be developed basically for the classification projects which are of a much smaller scale than those envisaged by the existing workflows. The phases of the workflow will be of much help to non expert users as they will be able to perform experiments with little or no guidance from the expert users.

2.7 Conclusion

This chapter started with a definition of machine learning followed by examples of machine learning applications in different fields. Five supervised classification techniques; decision trees, neural networks, k-nearest neighbour, support vector machines and random forests were then presented in section 2.4. These five

classification techniques will be used as sample techniques while doing experiments for the unknown settings for the proposed classifier workflow in the fourth chapter.

The later section discussed three most widely used workflows in machine learning and data mining projects. The existing workflows, CRISP-DM, the KDD process model and SAS-SEMMA are useful for large machine learning and data mining projects and hence do not provide clear procedural steps and tasks that an experimenter has to follow while creating models in a single application domain where the project may not be covered by the breadth provided by the workflow. The evaluations of the existing workflows have been discussed in section 2.7. Based upon the strengths and weaknesses of the three existing workflows, a new workflow will be presented in the fourth chapter.

A key stage in this new workflow will be classifier evaluation. The next chapter discusses classification evaluation methods. In order to select a classification technique options must be compared to each other. Each classification evaluation method has its bias depending on the application domain and the behaviour of the dataset that is going to be used for the experiments.

3 Classification evaluations

3.1 Introduction

While assessing and comparing performance of one learning algorithm over the other, accuracy and error rate are among the methods that are widely used. Other evaluation factors include speed, interpretability and risk when errors are generalised and ease of programmability (Craven 1996; Alpaydin 2004). This chapter describes more evaluations methods apart from accuracy and error rate. The discussion will be based on precision, recall, f1-score and ROC analysis that are used by machine learning researchers while comparing and assessing the performance of the classification techniques. The author also integrates machine learning and statistics by introducing statistical tests for the purpose of evaluating the performance of the classifier and comparison of the classification algorithms.

The author will use the strengths and weaknesses of the performance measures to recommend the appropriate measure(s) for the comparison of classification techniques. The output of this chapter will be the input to the fourth chapter where the proposed classifier workflow will be presented. The rest of this chapter is organised as follows. The chapter starts with performance measures in section two where accuracy, error rate, precision, recall, f1-score and ROC analysis are introduced. Section three provides five statistical tests followed by their evaluation from the author in the fourth section. Performance estimation methods will be discussed in the last section. The discussion considers the holdout method, k-fold cross validation and leave-one-out cross validation methods.

3.2 Performance measures

In machine learning and data mining, the preferred performance measures for the learning algorithms differ according to the experimenter's viewpoint (Bengio & Grandvalet 2004). This is much associated with the background of the experimenter as either in machine learning, statistics or any other field as well as an application domain where the experiment is carried out. In some application domains, experimenters' are interested in using accuracy and error rate while others precision,

recall and f1-score are of preference. This section provides the discussion of the performance measures used in machine learning and data mining.

3.2.1 Accuracy

Kostiantis (2007) defines accuracy as "*the fraction of the number of correct predictions over the total number of predictions*". The number of predictions in classification techniques is based upon the counts of the test records correctly or incorrectly predicted by the model. These counts are tabulated into a *confusion matrix* (also sometimes called contingency table) as shown in table 1with true class in rows and predicted class in columns. The confusion matrix shows how the classifier is behaving for individual classes.

TRUE CLASS	PREDICTED CLASS	
	YES	NO
YES	TP	FN
NO	FP	TN

Table 2: Confusion matrix for a two-case problem

TP Indicates to the number of positive examples correctly predicted as positive by the model.

TN Indicates the number of negative examples correctly predicted as negative by the model.

FP Indicates the number of negative examples wrongly predicted as positive by the model.

FN Indicates the number of positive examples wrongly predicted as negative examples by the model.

$$Accuracy = \frac{number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

Accuracy is a reasonable metric as long as the dataset remains evenly distributed (Zhong & Liu 2004). As most of the datasets used in our daily life are unbalanced, that is, there is an imbalanced distribution of classes; there is a need of having different classification evaluation factors for different types of datasets. Precision, recall, ROC analysis and the f1-score are the metrics which work well with unbalanced datasets (Manning & Schütze 1999).

Examples of unbalanced datasets can be found in network intrusion detection, direct marketing, web mining and medical diagnosis and in financial institutions while detecting risks in credit applications and detecting fraudulent credit cards (Lavrač et al. 2003). For the credit applications in financial institutions, it is a common practice that the number of customers who return their loans outweighs the number of customers who do not return their loans and number of non fraudulent credit cards outweighs the number fraudulent credit cards in fraudulent credit cards.

As a performance measure, accuracy only measures the number of correct predictions of the classifier and ignores the number of incorrect predictions. With this limitation, error rate was introduced to measure the number of incorrect predictions relating to the performance of the classifier.

Error rate

Mena (1999, p.138) comments that, *"for some applications, it is of interest to know how the system responded to the wrong answers and for what values of the condition attribute does this happens"*. . The error rate of the classifier can be used for such applications. Relating to the accuracy, the error rate of the classifier is just 1-Accuracy (M) on the training and test examples (Han & Kamber 2002).

$$Error\ rate = \frac{Number\ of\ incorrect\ predictions}{Total\ number\ of\ predictions}$$

False positives and false negatives are the two types of error rates. The application of these two error rates varies from one application domain to the other. Consider for example in medical diagnosis where false negative on a test for the serious disease causes a patient to go untreated and possibly risk life while false positives may lead to a second test which is more expensive.

Joachims (2002, p.9) justifies more weaknesses of using error rate as a performance measure in relation to other measures such as precision and recall. The weak justification of the error rate is related to the value 0 in relation to the precision and recall. It is a usual behaviour for the experimenters to think that the value 0 for the error rate means perfect precision and recall. However, the low error rate does not always mean perfect precision and perfect recall.

Accuracy and error rate are the correct performance measure(s) for the comparison of the classification techniques given balanced datasets.

3.2.2 Precision

In the area of information retrieval (IR) where datasets are much unbalanced, precision and recall are the two most popular metrics for evaluating classifiers (Manning & Schütze 1999; Fawcett 2004). Precision is used in many application domains where the detection of one class seems to be much more important than the other class such as in medical diagnosis, pattern recognition, credit risks and statistics (Provost, Fawcett & Ron Kohavi 1999). As an example, consider machine learning for fraud detection where the case of missing fraudulent transaction is quite different from the case of false alarm.

Precision measures the fraction of number of records predicted correctly by the classifier. It represents the proportion of selected items that the system got right (Manning & Schütze 1999) as the positive examples to the total number of true positive examples and false positives examples. In order to reduce the number of false positive or type II errors, the number of precision must be high (Witten & Frank 2000).

$$\text{Precision, } p = \frac{TP}{TP + FP}$$

3.2.3 Recall

Recall measures the fraction of positive examples correctly predicted by the classifier. It represents the proportion of the number of items that the system selected (Manning & Schütze 1999) as the positive examples to the total number of true positives and false negatives examples. Recall of the classification technique is supposed to be high in order to reduce the number of positive examples wrongly predicted as negative examples sometimes known as type I error.

$$\mathrm{Re}\,call, r = \frac{TP}{TP + FN}$$

Manning and Schütze (1999) assert the advantage of using *precision* and *recall* over accuracy and error rate. Accuracy refers to things got right by the system while error refers to things got wrong by the system. These two measures are not sensitive to any of the $TP, FP\ and\ FN$ values while Precision and recall are. There is a possibility of getting high accuracy while selecting nothing. Being surrounded by unbalanced dataset and the biasness of the accuracy and error rate on TP, FP and TN values; accuracy and error rate will be replaced by the use of precision and recall unless the dataset is really balanced.

3.2.4 F1-Score

In some applications, there is a tradeoff between precision and recall where as in selecting a document in information retrieval for example, one can get low precision but very high recall of up to 100% (Manning & Schütze 1999). Indeed, it is difficult to evaluate algorithm with high precision and low recall or otherwise. F1-Score combine precision and recall with equal importance into a single parameter for optimization and is defined as

$$F1 - Score = \frac{1}{\alpha \,{}^{1}\!/_{p} + (1-\alpha)\,{}^{1}\!/_{r}} \quad \text{(Manning \& Schütze 1999)}$$

Where *p* is precision, *r* is recall and α is the factor which determines the weighting between precision and recall. The value α = 0.5 is chosen for equal weighting of precision and recall and f1 measure is simplified to

$$F1 - Score = \frac{2 \times \Pr ecision \times \mathrm{Re} call}{\Pr ecision + \mathrm{Re} call}$$

F1-Score prefers results with more true positives while accuracy considers only a number of errors. With this bias, while evaluating classifiers some fields are not interested in only errors as measured by the accuracy but also finding interesting things even at the cost of returning some junk; in information retrieval systems (Manning & Schütze 1999). F1-Score suffers from the same problem as precision and recall (of using all column values of the confusion matrix), as it is a product of these measures.

The performance measure that overcomes the problem of using all the columns of the confusion matrix is ROC analysis which comprise of ROC graphs and ROC-AUC. Section 3.2.6 provides the discussion of ROC analysis where the area under the curve can be observed and then used for the comparison of the classification techniques.

3.2.5 Receiver Operating Characteristic (ROC) graph

Fawcett (2004) defines ROC graph as *"a technique for visualizing, organising and selecting classifiers based on their performance in a 2D space"*. Despite having several definitions, Fawcett's definition has been adopted for this dissertation as it shows directly where the technique is used and in which space. Originally conceived during World War II to assess the capabilities of radar systems, ROC graphs which uses area under the ROC curves abbreviated as AUC-ROC have been successful applied in different areas such as in signal detection theory to depict hit rate and false alarm rates, medical decision making, medical diagnosis, experimental psychology and psychophysics and in pattern recognition (Fawcett 2004).

The difference with the previous performance measures is that, ROC graphs are much more useful for domains with skewed class distribution and unequal classification

error costs (Fawcett 2004). With this ability, ROC graphs are much more preferred than accuracy and error rate. ROC graphs are plotted using two parameters; TP rate (fraction of true positives) or *sensitivity* which is plotted on the Y axis and FP rate (fraction of false positives) or *1-specificity* plotted in X axis as presented in figure 17. When several instances are plotted on a graph then a curve known as ROC curve is drawn (Kawahara 1999). The points on the top left of the ROC curve have high TP rate and low FP rate and so represent good classifiers (Winkler, Niranjan & Lawrence 2005).

True Positive Rate (TPR) or sensitivity

$$TPR = \frac{TP}{TP + FN}$$

True Negative Rate (TNR) or specificity

$$TNR = \frac{TN}{TN + FP}$$

To compare classifiers we may want to reduce the ROC performance to a single scalar value representing expected performance. The common method for reducing the ROC performance is to measure the area under the ROC curve abbreviated as AUC. After drawing the ROC curves of different classifiers, the best classifier is supposed to be nearby top left of the ROC curve. Figure 17 is an example of ROC graph for the comparison of three classifiers; SLN which is a traditional neural network, SVM and C4.5 rules

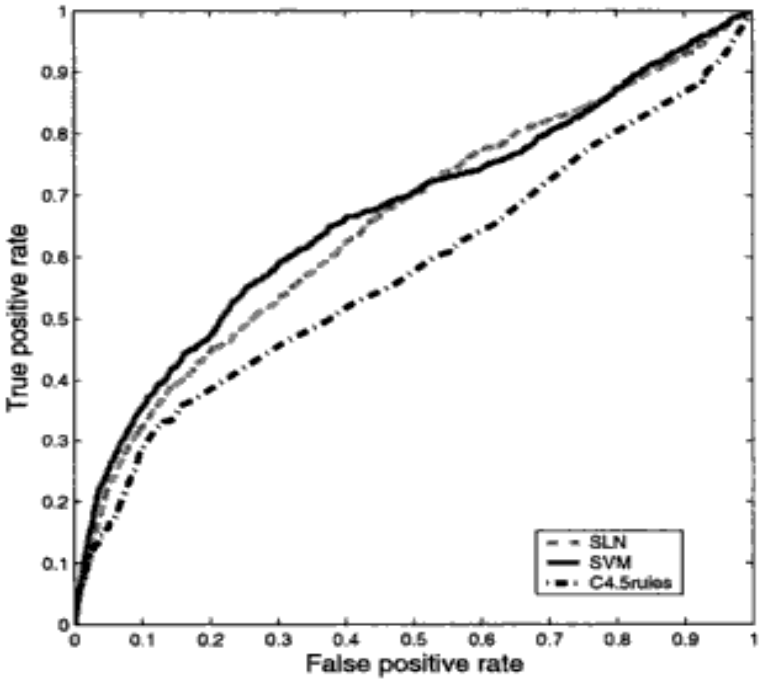

Figure 17: ROC curve for the comparison of three classifiers
(Source: (Winkler, Niranjan & Lawrence 2005))

3.3 Statistical tests

The classifiers induced by machine learning algorithms depend on the training set for the measurement of its performance. Statistical tests come into play when assessing the expected error rate of the classification algorithm or comparing the expected error rate of two classification algorithms. Though there are many statistical tests, only five approximate statistical tests for determining whether one learning algorithm outperforms another will be considered. This section provides the discussion about five statistical tests; Mc Nemar's, a test of the difference of two proportions, resampled paired t test, k-fold cross validated paired t test and the 5 x 2 cross validated paired t test. In the last subsection, the author provides an evaluation of these statistical tests based on the probability of type I error produced by the classifier.

3.3.1 Mc Nemar's Test

What is Mc Nemar's test?

Mc Nemar's test is a statistical test named after Quinn McNemar (1947) for comparing the difference between proportions in two matched samples and analysing experimental studies (Demuth 1999). It involves testing paired dichotomous measurements; "*measurements that can be divided into two sharply distinguished parts or classifications"* (Oxford English Dictionary 1989) such as *yes/no*, *presence/absence*, *before/after* (Demuth 1999). The paired responses are fabricated in a 2 x 2 contingency table and the responses are tallied in appropriate cells.

This test has been widely applied in a variety of applications to name a few; in marketing while observing brand switching and brand loyalty patterns for the customers (Beri n.d.), measuring the effectiveness of advertising copy or advertising a campaign strategy (Flynn 1986), studying the intent to purchase versus actual purchase patterns in consumer research (Foxall 2002), public relations, operational management and organisational behaviour studies and in health services (Osborn 2005).

Considering the application of McNemar's test in health institutions for example, where specific number of patients are selected at random based on their visits to a local clinic and assessed for a specific behaviour that is classified as risk factor for lung cancer. The classification of the risk factor is either present or absent. During their visits to the clinic they are educated about the incidence and associated risks for lung cancer. Six months later the patients are evaluated with respect to the absence or presence of the same risk factor. The risk factor before and after instructions can be tallied as tabulated in table 3 and evaluated using McNemar's test.

		Risk factor before instructions		
	Response 2	Response 1		Total
		Present	Absent	
Risk factor after Instructions	Present	e_{00}	e_{01}	$e_{00} + e_{01}$
	Absent	e_{10}	e_{11}	$e_{10} + e_{11}$
	Total	$e_{00} + e_{10}$	$e_{01} + e_{11}$	$e_{00} + e_{01} + e_{10} + e_{11}$

Table 3: matched paired data for the risk factors before and after instructions

Where e_{00}: The number of patients' that shows the *presence* of the risk factor for Response 1 and Response 2.

e_{01}: The number of patients' shows the *absence* of the risk factor for Response 1 and the *presence* of the risk factor for Response 2.

e_{10}: The number of patients' shows the *presence* of the risk factor for Response 1 and the *absence* for Response 2.

e_{11}: The number of patients' responded for the absence of the risk factor for Response 1 and Response 2.

$e_{00} + e_{01} + e_{10} + e_{11}$ represents the total number of examples in the test set.

Under the null hypothesis the change in risk factors; from presence to absence and vice versa should have the same error rates, which means $e_{01} = e_{10}$ (Dietterich 1998)

For McNemar, the statistic is as follows

$$x^2_{McNemar} = \frac{(e_{01} - e_{10})^2}{e_{01} + e_{10}} \text{ (Demuth 1999)}$$

In a 2 x 2 contingency table with 1 degree of freedom (1-column x 1-row), that is having one column and one row, the statistic for the McNemar test changes to

$$x^2_{McNemar} = \frac{\left(|e_{01} - e_{10}| - 1\right)^2}{e_{01} + e_{10}}$$

The null hypothesis would identify that there is no significant change in characteristics between the two times (as in table 2 for example, before and after instructions) (Demuth 1999). Thus we will compare our calculated statistic with a critical x^2, α with 1 degree of freedom or 3.84 (Osborn 2005). If the $x^2_{McNemar} > 3.84$, the null hypothesis is rejected and assumes a significant change in the two measurements.

Everitt (1992) comments on how to apply McNemar test for the comparison of the classifiers. Having available sample of data *S* divided into training set and testing set, both algorithms A and B are trained on the training set which results in two classifiers P_1 and P_2. These two classifiers are then tested using the test set. The *contingency table*, provided in table 4, is used to record how each example has been classified.

e_{00} **Number of examples correctly classified by both**	**e_{01}** **Number of examples misclassified by algorithm 1 but not 2**
e_{10} **Number of examples misclassified by algorithm 2 but not 1**	**e_{11}** **Number of examples misclassified by both**

Table 4: Contingency table for the comparison of the classification techniques (Adapted from: (Alpaydin 2004))

If the null hypothesis is correct then, the probability that the value for the x^2 with 1-degree of freedom is greater than 3.84 is less than 0.05 and the null hypothesis may be

rejected in favour of the hypothesis that the two algorithms have different performance measurements when trained in a particular training set.

Dietterich (1998) comments on the advantage of using this test compared to other statistical test as such Mc Nemar's test has been yielded to provide low type 1 error. Type 1 error means *ability to incorrectly detect differences while there is no difference that exists* (Dietterich 1998). Despite having aforementioned advantage, this test is associated with several problems. Firstly, a single training set is used for the comparison of the algorithms and hence the test does not measure the variations due to the choice of the training data (Dietterich 1998).

Secondly, Mc Nemar's test is a simple holdout test, where by having available sample data; test can be applied after the partition of the data into training set and testing set. For the comparison of the algorithms, the performance is measured using the training data rather than the whole sample of data provided. Mc Nemar's test as a performance measure for the comparison of the algorithms from different application domains has been associated with the aforementioned shortcomings.

These shortcomings have resulted into the growth of other statistical tests for ML classification techniques, include *a test for the difference of two proportions*, *the resampled t test*, *k-fold cross validated t-test* and *5 x 2 cv paired t test*.

3.3.2 A Test for the Difference of Two Proportions

A test for the difference of two proportions measures the difference between the error rate of algorithm A and the error rate of algorithm B (Dietterich 1998). Consider for example, P_A be the proportion of the test examples incorrectly classified by algorithm A and P_B be the proportion of the test examples incorrectly classified by algorithm B,

$$P_A = \frac{e_{00} + e_{01}}{e} \quad , \quad P_B = \frac{e_{00} + e_{10}}{e}$$

The assumption underlying this statistical test is that when algorithm A classifies an example *n* from test set the probability of misclassification is P_A . Hence, the number of misclassification for *n* test examples is a binomial distribution with mean nP_A .

This statistical test is associated with several problems, firstly as P_A and P_B are measured on the same test set, they are not independent. Secondly, the test does not measure the variations due to the choice of the training set or the internal variation of the algorithm (Dietterich 1998). Lastly, this test suffers with the same problem as McNemar test; does not measure the performance of the algorithm in the whole dataset (with all sample size) provided; rather it measures the performance on the smaller training data after partition.

3.3.3 The Resampled Paired *t* Test

With this statistical test, usually a series of 30 trials is conducted (Dietterich 1998). In each trial, the available sample data is randomly divided into training set of specified size and testing set. Learning algorithms are trained on the training set and the resulting classifiers are tested on the test set. Consider, P_A *and* P_B be the proportion of test examples misclassified by algorithm A and algorithm B respectively. For the 30 trials we will result into having 30 differences

$$P^i = P_A^{(i)} - P_B^{(i)} \text{ (Dietterich 1998)}$$

Among the potential drawbacks of this approach is, the value of the differences (P^i) are not independent because the training and testing sets in the trials overlap.

3.3.4 The k-fold cross validated Paired t test

The k-fold cross validated paired t test was introduced to overcome the problem underlined by the resampled paired t test; overlapping of the trials. This test works by dividing the sample size into *k* disjoint sets of equal size $T_1 \cdots T_k$ and then k trials are conducted. In each trial, the test set is T_i and the training set is the union of all the other sets.

This approach is advantageous as each test set is independent of the others. However this test suffers from the problem that the training data overlap (Dietterich 1998). Consider for example, when k=10, in a 10-fold cross validation, each pair of the training set shares 80% of the examples (Alpaydin 2004). This overlapping behaviour may prevent this statistical test from obtaining a good estimate of the variation that would be observed if each training set were completely independent of the previous training sets.

3.3.5 The 5 x 2 cross validated Paired t Test

With this test, 5 replications of the twofold cross validation are performed (Alpaydin 2004). In each replication, the available data are partitioned into two equal sized sets, lets say S_1 *and* S_2. Each learning algorithm is trained on one set and tested on the other set and this results into four error estimates as shown in figure 18.

The choice of the number of replications is not the responsibility of the experimenter; this is how the test requires. The test allows the applications of only five replications in a twofold cross validation as exploratory studies shows that, the use of more or less of five replications increases the risk of type I error which is supposed to be low for the betterment of the test (Dietterich 1998)

This test has one disadvantage, in each fold the training set equals the testing set and hence results into learning algorithms to be trained in training sets half the size of the whole training sets (Dietterich 1998). For better performance of the learning algorithm, there supposed to have a large training set than the testing set.

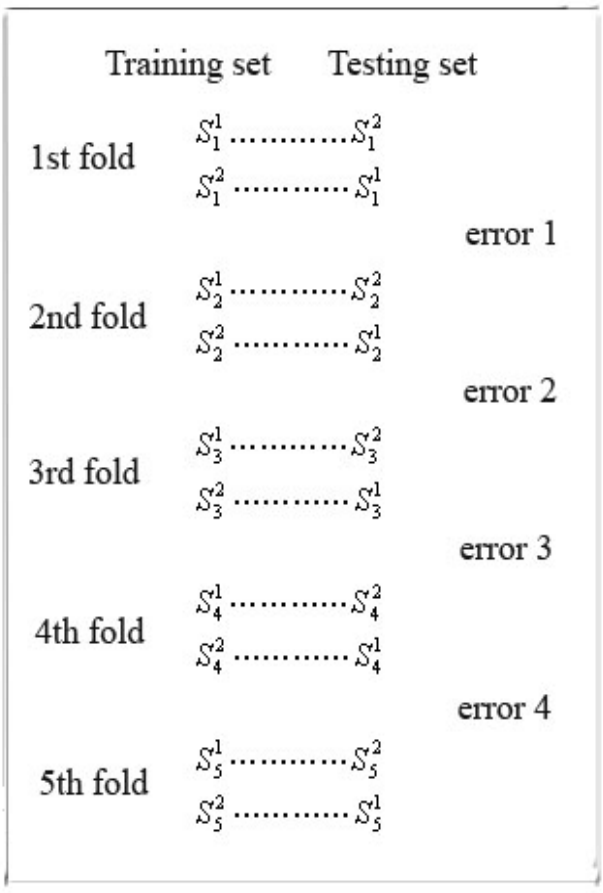

Figure 18: 5 x 2 cross validation (Adapted from Alpaydin 2004)

3.4 Statistical tests evaluation

The statistical tests discussed in section 3.3 can be evaluated using the probability of type I error. Type I error or false positives as shown in table 4 refer to the ability of the statistical test to detect algorithms difference when in reality no difference exists (Field 2005). Type I error is also refereed to as α level while type II error is refered to as β level (Berg & Latin 2007). For the difference between the two errors consider for example, as shown in table 5, when a woman goes to the hospital trying to find if she has cancer or not. When a system detects that a woman is suffering from cancer while not, this is false positives or type I error. If a system reports *"negative"* when the woman is infact suffering from cancer this is false negatives or type II error.

TEST RESULTS	ACTUAL CONDITION	
	Infected	Not infected
Infected	True Positives	Type II error (False Negatives)
Not infected	Type I error (False Positives)	True Negatives

Table 5: Type I and Type II errors.

The discussed statistical tests can be evaluated using values depicted in figure 19 where by the probability of type I error has been measured against each statistical test. The alpha level or threshold for the probability of the type I error has been set to 0.05. According to Dietterich (1998) the resampled paired t test has the highest probability of type I error compared to the rest of the tests. Its probability for the type I error is almost 0.25 followed by the test for the difference of two proportions which is near by 0.1.

K-fold cross validation or XVal as shown in figure 19 is the third statistical test in the ranking of the probability of type I error with more than 0.05.The Mc Nemar and 5 x 2 cv paired t test are the only two statistical tests that provide less than 0.05 probability of type I error which is the required threshold. These two tests require other evaluation factor apart from the probability of the type I error as they both fall below the required threshold.

The comparison of type I error probability for the five statistical tests have resulted into two statistical tests McNemar test, as discussed in section 3.3.1 and 5 x 2 cross validated paired *t* test which uses the resampling method of the data discussed in section 3.3.4. The choice of the best statistical test between the two is determined by the computational cost of running the learning algorithm (Dietterich 1998). The 5 x 2 CV test have been characterised to have high computational costs for the algorithms that are going to be executed only once while McNemar test have been characterised as the test associated with the low type I error. This comes into a conclusion that for small scale classification projects, McNemar test have to be applied as there is no

need of executing the learning algorithms 10 times and hence increasing the rework rate which the developed classifier workflow intends to reduce.

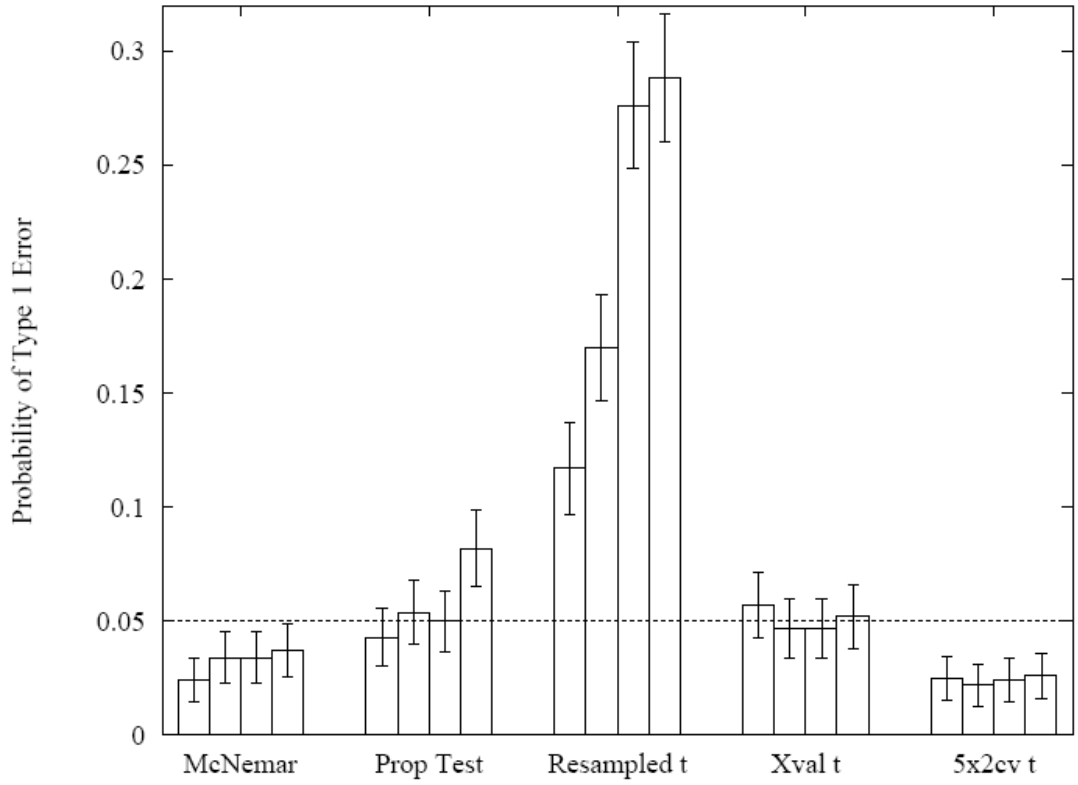

Figure 19: Probability of type I error for five statistical tests (Dietterich 1998)

3.5 Performance estimation methods

Substantial research has been devoted to the development and analysis of the algorithms for building classifiers and comparing classification algorithms. Despite having several performance measures; classification accuracy and error rate, by far, are regarded as the commonly used performance metrics. Subtle estimators of these performance measures have been developed such as cross validation and a variety of bootstrap method. This section provides the discussion of three common performance estimators used in machine learning and data mining projects; holdout test method, k-fold cross validation and leave-one-out method.

3.5.1 Holdout test

The holdout test or sometimes called test set (Craven 1996) estimation works by randomly dividing data into two mutually exclusive subsets; training set and testing

set or holdout set (Ron Kohavi 1995; Micheli-Tzanakou 1999). Two-third (2/3) of all data is commonly designated for the training and the remaining one-third, 1/3, for the testing of the classifier. The reserved training set is given to the classifier, and the learning algorithm is tested using the test set.

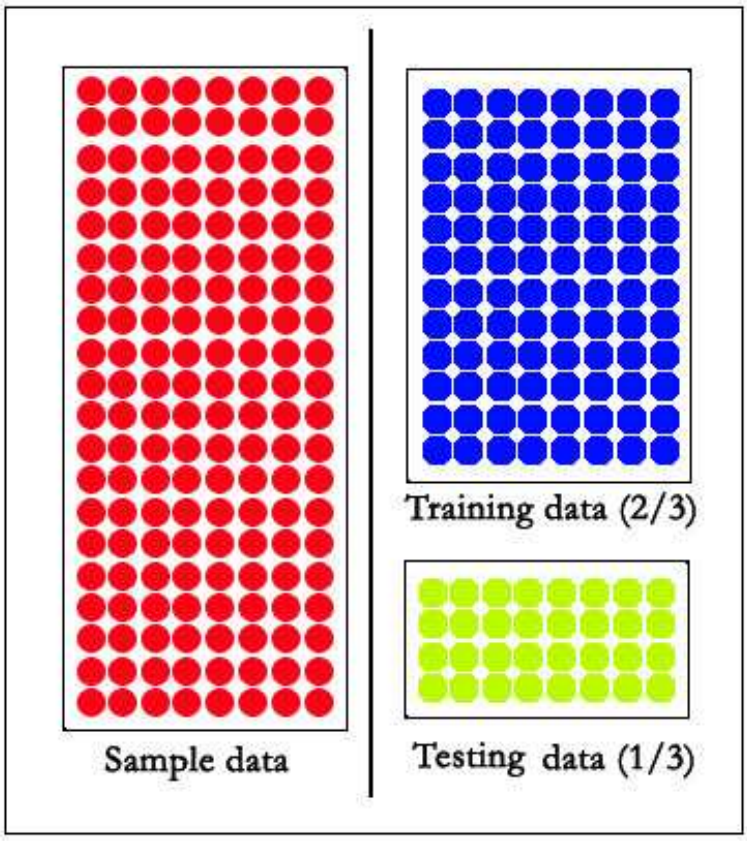

Figure 20: Process of dividing data into training set and testing set using the holdout method

Hold out method works differently compared to random sampling methods such as cross validation. In random sampling; for different partitions, holdout method is repeated *k* times and the accuracy is estimated by averaging the accuracies obtained from each holdout (Kohavi 1995).

Kohavi (1995) outlined the biasness of the method while dividing the data into training and testing. The more instances are left for test set, the higher the bias of estimate; however fewer instances for the test set the wider confidence interval for the accuracy.The hold-out technique does not account for the variance with respect to the training set, and may thus be considered inappropriate for the purpose of algorithm

comparison (Dietterich 1998). Moreover, it makes an inefficient use of data which inhibits its application to small sample sizes only (Bengio & Grandvalet 2004)

3.5.2 K- Fold Cross Validation (CV)

In K-fold cross validation the available data is partitioned into *k* separate sets of approximately equal size (Craven 1996). The cross validation procedure involves *k* iterations in which the learning method is given *k-1* as the training data and the rest used as the testing data. Iteration leaves out a different subset so that each is used as the test set once (Craven 1996).

Cross-validation is a computer intensive technique, as it uses all available examples in the dataset as training and test sets (Bengio & Grandvalet 2004). It mimics the use of training and test sets by repeatedly training the algorithm k times with a fraction $1/k$ of training examples left out for testing purposes. It is regarded as the kind of the holdout test estimate.

With this strategy it is possible to exploit much larger dataset compared to leave-one-out method in section 3.5.1. However, since the training and testing is repeated *k* times with different parts of the original dataset, it is possible to average all test errors (or any performance measure used) in order to obtain a reliable estimate of the model performance on the newly test data (Nelles 2001).

The advantage of this test is that each test set is independent of the others (Dietterich 1998). Due to its processes, K –fold cross validation suffers from the problem of the overlapping of the test sets (Dietterich 1998). This makes the K-Fold Cross Validation to be termed as the test which lacks computer efficiency (Bengio & Grandvalet 2004). In a 10-fold cross validation, each pair of the training sets shares 80 percent (80%) of the examples. This overlapping may result into failure of obtaining good estimates of the amount of variation.

3.5.3 Leave-one-out cross validation

Leave-one-out cross validation is refereed to as *n*-fold cross validation where *n* is the number of instances (Witten & Frank 2000). Given the dataset with *n* cases, one

observation is left out for testing and the rest *n-1* cases for training (Tang et al. 2004). Each instance is left out once and the learning algorithm is trained on all the training instances. The judgement on the correctness of the learning algorithm is based on the remaining instances. The results of all *n* assessments, one for each instance, are averaged and the obtained average represents the final error estimate of the classifier.

Leave-one-out method is attractive in a number of reasons. Firstly, there is a greatest possible amount of data which is used for training in each case, this increases the possibility that the classifier is the accurate one (Witten & Frank 2000). Secondly, the method tends to simply repetition which is performed in the k-fold cross validation (repeated 10times 10-fold cross validation, for example) as the same results are obtained every time.

Despite having the simplicity in operation, leave-one-out cross validation is associated with several disadvantages. The method is associated with the computation cost. Considering, for the entire learning algorithm we have *n* instances then the learning procedure must be executed *n* times and this is quite infeasible in large datasets (Witten & Frank 2000). Also the method can not be stratified as there is only single example reserved for the test set. Stratification means *getting correct proportions of examples in each class into the test set* (Alpaydin 2004).

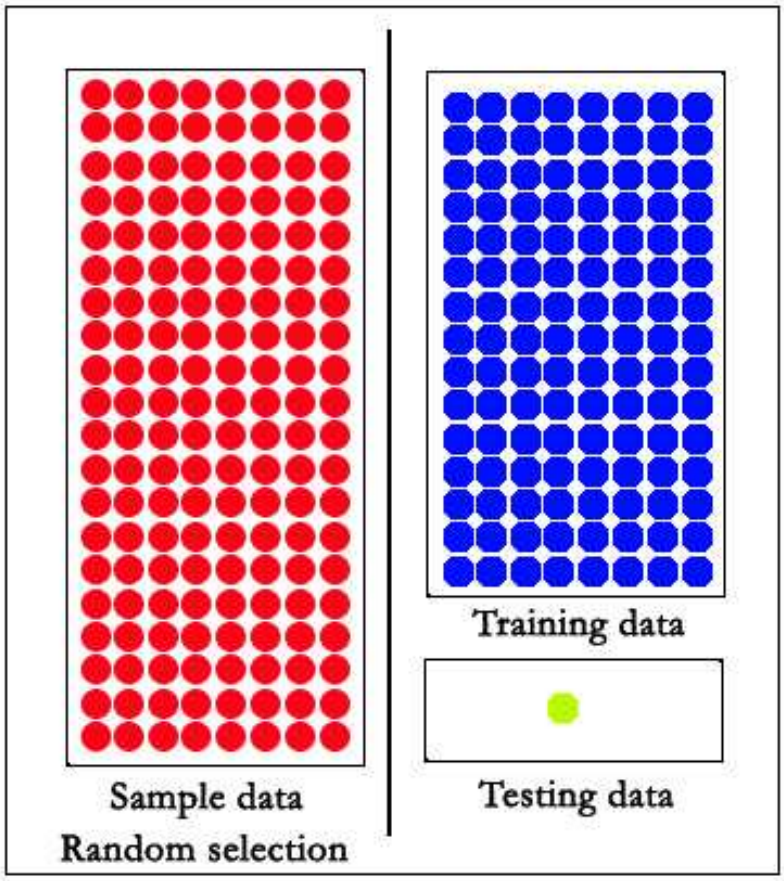

Figure 21: Process of randomly selecting a data sample for use in the test set with the remaining data going towards training.

3.6 Conclusion

The aim of this chapter was to introduce the classification evaluation methods that are applicable in machine learning projects. The chapter was divided into five sections where it was introduced in section 3.1. In the later section 3.2, performance measures were discussed. The discussion was on accuracy, error rate, precision, recall, f1-score and ROC analysis. The performance measures were discussed together with their advantages and disadvantages.

In section 3.3, five statistical tests for assessing and comparing the performance of the classification algorithms were introduced. The discussion was based on McNemar's test, a test for the difference of two proportions, resampled paired t test, k-fold cross validated paired t test and the 5 x 2 cross validated paired t test. The strengths and weaknesses of each statistical test were succinctly discussed.

The evaluation of the statistical test based on their probability of type I error was presented on the fourth section. The aim of this section was to measure the probability of each statistical test to incorrectly detect differences while there are no differences that exist. After the evaluation, McNemar and k-fold cross validated paired t test had the probability of less than 0.05; evaluated using other biasness McNemar outperformed k-fold cross validated paired t test.

The last section provided the discussion of three mostly used performance estimators which works hand in hand with the performance measures. Holdout test method was introduced first followed by k-fold cross validation and then leave-one-out method. This chapter plays an important role of acting as the input to the next chapter where the proposed classifier workflow will be introduced.

4 Proposed classifier workflow

4.1 Introduction

Chapter 2 and 3 introduced the underlying concepts and prior workflows that have been developed by data mining researchers while trying to build models and compare classifiers to be used in multiple domains. The classifier workflow proposed in this chapter is not the final version, experiments need to be carried out and the evaluation needs to be done before the final version of the workflow. These experiments will be discussed in chapter 5, with the final workflow presented in chapter 6.

This chapter has been divided into four sections. The proposed classifier workflow is presented in the next section followed by its phases in different layers in section 4.3. The inner layer comprises of experimental design, algorithm selection, preprocessing, performance estimation method selection, performance measures and algorithm parameters and the outer layer with experimentation and evaluation. From its phases, there are unknown settings such as, dataset threshold, performance estimation method and algorithm parameters that needs to be set using the experiments in chapter 5 and this forms the last section of this chapter.

4.2 Classifier workflow overview

The proposed classifier workflow as shown in figure 22 has been divided into two layers; inner layer and outer layer. The inner layer comprises of six workflow phases while the outer layer comprise of two phases. The phases of the classifier workflow are iterative as the iteration as discussed in section The experimenter is expected to start from the inner layer where the steps start from experimental design, algorithm(s) selection, preprocessing, performance estimation method selection, performance measures to algorithms parameters. After algorithm parameters, the experimenter is expected to be ready to perform the experiment and hence the next step is to shift to the outer layer of the workflow where there are two steps; experimentation and evaluation. Figure 24 presents the proposed classifier workflow.

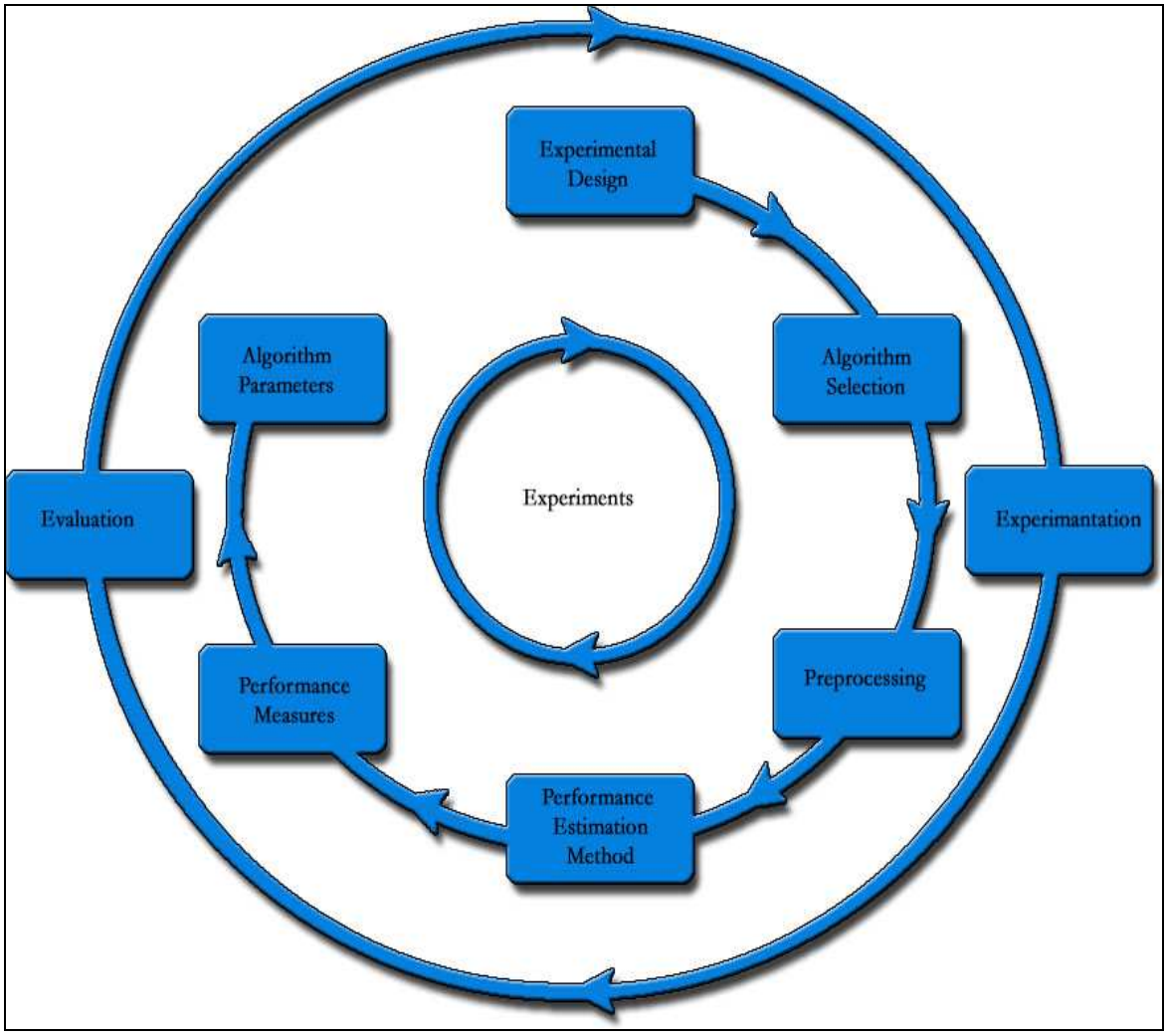

Figure 22: Proposed classifier workflow

The arrows in the classifier workflow in figure 22 indicate the starting and finishing points of the phases and the dependencies between them. The inner circle represents the core phases before experiments and the outer circle represent a repeatable phase. Consider for example, if an experimenter needs to perform certain experiment for an application domain, the experiment needs to start at the inner circle where all the settings will be established before using the outer circle to perform several experiments and then evaluate the results.

4.3 Classifier workflow phases

The classifier workflow provided in section 4.1 of this chapter has eight phases which need to be followed while building classifiers or comparing classification algorithms in a single application domain. The first phase is the experimental design where the

objectives of the experiments are set followed by algorithm selection in the second phase. With algorithm selection, the experimenter has to evaluate algorithms depending on the algorithms' core characteristics and capabilities such as interpretability, training speed, testing speed and so on. The result of this phase is the algorithm(s) that fit for an application domain where the objectives have been set and experiments have to be carried out.

The third phase is preprocessing, where, the experimenter has to find a way to deal with noisy, missing and inconsistent values in the datasets. The fourth phase involves the selection of the performance estimation method; this is relatively related to the sample size of the dataset that is going to be used in the experiment. As discussed in section 3.5; there are three types of performance estimation methods namely, holdout test, k-fold cross validation and leave one out estimation. Experimenter has to select the performance estimation method in relation to the dataset characteristic and the performance measure(s) which are going to be used in algorithms evaluation.

The fifth phase involves the selection of the performance measures. In this dissertation, performance measures have been divided into two categories, normal performance measures and statistical tests. For the normal performance measure, experimenter has a range of choices from accuracy, error rate, precision, recall and f1-score to ROC analysis. For the statistical tests based on the evaluation described in section 3.4 only McNemar's test for assessing and comparing classification algorithms is available due to its probability of producing low type I error. The sixth phase involves setting parameter(s) for the selected algorithms. The last two phases involves experimentation and evaluation.

4.3.1 Experimental design

Machine learning and data mining classification projects require very carefully thought about experimental design. If this phase has not done properly the comparative study of classification algorithms can result in statistically invalid conclusions (Salzberg 1999). As discussed in section 2.6.4, the first important step in machine learning is the definition of objectives. Despite having different name in relation to existing workflows, experimental design serves the same purpose.

Antony (2003) defines experimental design as '*the laying out of a detailed experimental plan in advance of doing the experiment"*. It involves the process of planning, designing and analysing the experiment so that valid conclusions can be drawn efficiently and effectively (Antony 2003). Before starting to experiment for a classifier related to a single application domain, different fields have different basic needs from the classifier(s) and hence objectives need to be set. These objectives will act as the guideline for the experimenter to follow while experimenting.

Considering financial institutions such as banks and supermarkets for example, one field will need a classifier that provides explanation and other will need a classifier with faster training speed. Different objectives need to be set for different application domains. The experimenter using the classifier workflow has a responsibility of setting these objectives in experimental design phase so as the predicted outcome of the experiment has to be known before committing further to the experiment.

The aim of performing designed experiment is to reduce the rework rate, to reduce model development time and to improve the functional performance of the models (Antony 2003). Without knowing the objectives of your experiment, there is a big chance of doing duplicated work when the objectives have been achieved, experimental design will reduce the rework rate. Also having the objectives set will improve the functional performance of the models and reduce the model development time.

4.3.2 Algorithm selection

The second phase starts after finishing the first phase; experimental design where the objectives of performing the experiments were set. This phase involves selection of algorithms where by various algorithms related to an application domain are evaluated using the classification algorithm evaluation table shown in table 2. Experimenter is supposed to know abilities and disabilities of different classification techniques. The most important factors for an application domain will be listed followed by the least important factors.

Consider table 6 for example, if an application domain requires explanation then column 1 will be the most important factor for the experimenter to consider before evaluating the training and testing speed of the classifiers. With this evaluation from the beginning, experimenter will do the experiments using only required classifiers as classifiers that do not fit for such an application domain will be eliminated.

Algorithm	Classification algorithm selection criterion		
	Interpretability	Training Speed	Testing Speed
Decision Tree	✓	X	✓
Neural Network	X	X	✓
KNN	✓	✓	X
SVM	X	X	✓
Random Forest (RF)	X	X	✓

Table 6: Classification algorithm evaluation table

The classification algorithms selection table will allow the experimenter to evaluate the algorithms that is going to be used in the experiment if they fit in an application domain due to objectives set in the first phase. Classification algorithms are plotted, in the left hand side against classification algorithm selection criterion at the top right part of the table. The arrangement of the classification algorithm selection criterion is from the most important factor at the top left part to the least factor. Two symbols will be used to indicate the applicability of the classification algorithm selection criterion in relation to the algorithm; tick and cross symbol.

The tick symbol indicates applicability of the criteria while cross will be used for non-applicability criteria. In table 6 for example; with interpretability selection criteria, decision tree and KNN have the tick symbols while others don't, this is because from the five listed algorithms, only these two algorithms tend to provide explanation. If the experimenter is developing the classifier where the main factor is interpretability, only decision trees and KNN will be used and other algorithms will

be eliminated. This will allow the experimenter to reduce the work that needs to be done by performing experiments with required algorithms only.

Classification algorithm selection table: description

This section provides the discussion from the research literature on the abilities and disabilities of the learning algorithms provided in table 6.

From the characteristics of the decision tree discussed in section 2.5, one advantage that this algorithm provides is interpretability through *if-then* rules. Also the algorithm uses divide-and-conquer strategy while training instances, this result into the training speed of the algorithm to be slow. The newly instances are classified using the model which have been developed and results into faster testing speed.

The discussion of the artificial neural network has been presented in section 2.6.2, one of its characteristic is that it requires inputs and provides output hence it is regarded as the black box processing. This algorithm does not provide any explanation which can be understood by expert or non-expert users. The artificial neural network like the decision tree develops models before classification of the new instances, this result into slow training speed and faster testing speed.

KNN has the ability of providing explanation which can be interpreted by its users. This algorithm classifies new instances by measuring the distance to its nearest neighbour. As discussed in section 2.6.3, with this algorithm, there is no model which is developed beforehand, when a new instance arrives only distance is measured and hence resulting into fast training speed. The distance is measured until when the instance needs to be classified, this results into slow testing speed as there is no prior model which is developed.

With support vector machines, Hornberg (2007) presented the advantage of using this learning algorithm over the MLP as the training time is shorter. But the overall training speed of the algorithm is slow when compared to other algorithms such as KNN. For the testing speed, the testing speed of the SVM depends on the number of support vectors which usually are not many. Therefore SVM are considered to

provide slow training speed and faster testing speed while it is not capable of providing explanation.

Random forest which is made up of random trees does not provide explanation for the experts and non-experts. Rather the learning algorithm using the ensemble of trees as discussed in section 2.6.5 has been characterised to provide slow training and fast testing speed.

4.3.3 Preprocessing

Today's real world datasets are associated with noisy, missing and inconsistent values due to rapid emergence of data collection and processing tools (Han & Kamber 2002; N. Zhang & Lu 2007). It is well known that more than eighty percent (80%) of time in machine learning and data mining projects is spent on data preprocessing which lays the basis for the experiments (N. Zhang & Lu 2007). These three concepts; noisy data, missing data and inconsistent data in general are termed as data problems (K. Narita & H. Kitagawa 2006). Noisy is related to data that contain incorrect values or outliers from expected. Missing or incomplete data means data that are lacking attribute values or certain attributes of interest while inconsistent values are the values that seem to be different from one source to the other.

Classification algorithms require preprocessed data where by noisy, missing and inconsistent values have been removed for better classifier's performance. There are a number of ways available for performing the preprocessing step, namely data cleaning, data integration and transformation and data reduction (Han & Kamber 2002). Han & Kamber (2002) comment that when a data preprocessing step is performed prior to classifier development, can improve the overall quality of the classifier or reduce the time required for the classifier development.

Data cleaning is unbiased and hence works for all the data problems; filling in missing values, smoothing noisy data or removing outliers and resolving inconsistencies. Han & Kamber (2002) address a number of ways for dealing with the missing values in datasets. In this dissertation these methods have been categorised into two groups. The first group contains ignoring the tuple when the class label is missing, filling in

missing values manually and using the global constant such as *"unknown"* to fill in missing values. The first method is not very effective unless there is more attributes that are missing apart from the class label. The second method is infeasible if working with large dataset and the last method will be disadvantageous if there is many unknowns as this will result into the algorithm classify the unknown constant thinking, it is an interesting concept.

The second group include methods such as using the attribute mean to fill in missing values, using the attribute mean to fill in missing values for all samples belonging to the same class as the given tuple and using the most probable value to fill in missing values (Han & Kamber 2002). With the first method, considering for example, you have some missing values in the *salary* attribute and the average salary is €2000, then this value will be used for all the salary missing values. For the second method, consider classifying customers according to the credit_risk replace the missing values with the average income for customers falling into the same credit risk category as that of the given tuple.

4.3.4 Performance estimation method selection

This phase involves selection of performance estimation method. As discussed in section 3.3 there are three types of the performance estimation methods available, these are the holdout method, k-fold cross validation and leave one out cross validation. Choice of technique is determined by the size of the dataset being used. Experimenters have to know the threshold for the sample size of data as to what amount the dataset will be regarded as small dataset or large dataset. If the amount of data is large enough then holdout method will be appropriate for evaluation. If the provided dataset is small then k-fold cross validation is the appropriate approach, otherwise for very small datasets leave-one-out method is the appropriate approach.

4.3.5 Performance measures

Several performance measures and their strengths and weaknesses have been discussed in section 3.2 namely accuracy or error rate, precision, recall, f1-score, ROC curve and Mc Nemar's test. For this phase, one or more factors need to be set so that the classifier will be tested using that performance measure. Precision and recall

seem to provide some bias in the type of dataset that they can work effectively. From the literature provided in chapter 2; accuracy, f1-score, ROC analysis and Mc Nemar's test are the methods from statistics and normal performance measures respectively that work with datasets with little bias comparing with associated factors. Accuracy or error rate and ROC graph using AUC will be used for balanced datasets; while f1-score which is a product of a precision and recall will be used for unbalanced datasets. McNemar test will be considered as the performance measure when the dataset is unbalanced and paired.

After selecting the performance measure(s) for the comparative study, the results needs to be tabulated as in table 7.

Performance Measures	Algorithm	
	Algorithm A	Algorithm B
Accuracy		
Precision		
Recall		
F1-Score		

Table 7: Performance measures table for comparing two or more algorithms

The performance measures are placed on the left hand side while two or more algorithms that need to be compared are placed at the top right hand side. The respective performance measure(s) will be recorded against each algorithm. If the appropriate performance measure is accuracy for example, then its respective value must be recorded against each learning algorithm.

4.3.6 Algorithm parameters

Different parameter settings in different algorithms tend to provide different results for the evaluation phase in classification comparative studies for machine learning and data mining. For the classifier workflow, an experimenter is expected to set not more than two key parameters for each classification technique where each parameter

has three possible values: low, medium and high. These values have been chosen randomly to represent these three values. For the naïve users in machine learning and data mining, they can use the table together with their suggested parameter values (low, medium and high) while experts can use the frame of the table to input parameters together with their respective values. With these three values an experimenter can evaluate if the classifier works well with what parameters. For example in k-nearest neighbour, different values of *k* tend to provide different results for the performance measures. Obtained results, needs to be plotted in a table for clearer way of comparison.

Table 8 represents a table for setting parameter values against algorithms. This empty table can be used by the expert user as it will tough for non-experts.

Algorithm	Parameter					
	Parameter 1			Parameter 2		
	Low	Medium	High	Low	Medium	High
Decision Tree	(Parameter name)			(Parameter name)		
Neural Network						
KNN						
SVM						
Random Forest						

Table 8: Table for testing parameter values against classification algorithms

With three parameter values set as low, medium and high, it will be easy to notice what values do the classification algorithm provides better performance. For the

classification algorithms and parameter values, performance measures discussed in section 3.2 needs to be high. If the algorithm's performance is affected by two key parameters, then the second column might be used inserted in columns as parameter 1 and 2.

4.3.7 Experimentation

This is the seventh phase for the whole classifier workflow and the first phase in the outer circle part of the classifier. Experimentation is adopted after setting-up one or more key algorithm parameters for a classification technique. The two steps that make up the outer circle of the workflow are repetitive as such when an experimenter knows most of the settings from the inner circle phases the remaining work is only changing the values for each classification techniques. After the experimentation, the results need to be evaluated.

4.3.8 Evaluation

This is the last phase of the classifier workflow where the results from experiments need to be evaluated. The overall results obtained in the experiments and tabulated in table 9 where classification algorithms plotted against performance measure requires evaluation. With this final phase, Experimenter has to add one row which includes statistical test for assessing the significance of the difference between algorithms.

Performance Measures	Algorithm	
	Algorithm A	Algorithm B
Accuracy		
Precision		
Recall		
F1-Score		
McNemar's Results		

Table 9: Algorithm evaluation table with statistical tests

4.4 Unknown settings

From the eight phases of the proposed classifier workflow discussed in section 4.3, some phases show incompleteness which needs to be addressed for the classifier workflow to work efficiently and effectively. This section introduces questions which need to be answered before the final version of the proposed workflow which will be provided in chapter 6.This section will serve as the starting point for chapters 5 and 6.

4.4.1 Dataset threshold

Different datasets have different sample sizes in relation to the number of attributes and number of instances. With the proposed classifier workflow and its phases in section 4.1 and section 4.2 respectively there is no limit which has been set on the categorisation of the datasets in relation to the number of features and number of instances. This categorisation is sometimes known as the dataset threshold. With the dataset threshold being set, an experimenter will use it to decide if the dataset is either small or large by looking at the number of instances. Knowing the size of the dataset would simplify the process of choosing the performance estimation method. For the classifier workflow, setting the threshold of the dataset is a problem which needs experimentation.

4.4.2 Performance estimation

There is no clear discussion as to when an experimenter is supposed to use each of the performance estimation methods. As discussed in section 3.3, there are three performance estimation methods; these are holdout method, k-fold cross validation and leave-one-out method but only two methods are mostly used k-fold cross validation and leave-one-out method. These criterions are mostly associated with the dataset threshold. Knowing the number of instances and the number of features in the dataset, there is a very good chance to select performance estimation method that fit for that dataset.

Holdout method works with very large datasets while k-fold cross validation works with medium to large datasets. Leave-one-out method works well with small datasets. Knowing where each performance estimation method works perfectly is not the

solution to the classifier workflow. The dataset threshold needs to be set and the performance estimation method in relation to such dataset needs to be evaluated using the *if-then* rules. The results of the sample size threshold performance estimation method are supposed to look like as shown in figure 23.

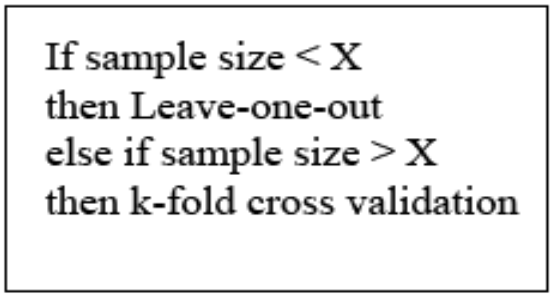

Figure 23: Dataset threshold for the performance estimation methods

4.5 Conclusion

This chapter aimed at introducing the classifier workflow that can be used in machine learning and data mining projects. The chapter was divided into five sections. The classifier workflow was introduced before its presentation on the second part of the chapter. The third section provided discussion about the eight phases of the classifier workflow namely; experimental design, algorithm selection, preprocessing, performance estimation methods selection followed by the performance measures. Other phases include algorithm parameters settings, experimentation and evaluation. Unknown settings from the phases of the classifier workflows were introduced in the last part of the chapter.

The next chapter proceed from what was left out in section 4.4, that is, unknown settings of the proposed classifier workflow together with the key parameters for each of the classification technique discussed in section 2.5. The chapter provides the experiments for setting-up unknown settings for the complete classifier workflow that will be provided in chapter 6. The result of chapter 5 plus the proposed classifier workflow in chapter 4 contributes to the complete classifier workflow.

5 Experiments

5.1 Introduction

This chapter is aimed at establishing benchmark settings described in section proposed classifier workflow in chapter 4. To decide on how different measures have to be set for the classifier workflow, experiments have to be performed on a selection of datasets from UCI machine learning repository. This chapter describes these experiments. Different datasets with different amount of examples created randomly will be used together with k-fold or leave one out cross validation as the classification evaluation method. The performance of the technique will be measured using f1-score. The results obtained will act as the input to the proposed workflow.

These experiments have been divided into two parts as discussed in section 4.4. The first part provides the experiments for setting the dataset threshold followed by the experiments for selecting the performance estimation methods. The results from experiments taken in this chapter, together with the proposed classifier workflow, will contribute to the classifier workflow in chapter 6.

5.2 Dataset threshold

As presented in section 1.1, dataset threshold refers to the maximum number of instances that should be considered applicable for performance estimation method. Considering the proposed classifier workflow in chapter 4, there is no number of instances that have been suggested as the dataset threshold. The main aim of this experiment is to establish minimum number of instances that the will help on deciding the dataset threshold. These experiments involve the use of different dataset together with three performance estimation methods and decide if the dataset as small, medium or large.

Three performance estimation methods have been discussed in section 3.5 namely, holdout method, k-fold cross validation and leave one out cross validation. The holdout method has been identified to work well on very large datasets, but nothing has been identified for the remaining two estimators. This experiment will be

performed using the performance estimators and the dataset which has been created randomly. The accuracy of the dataset with all instances will be regarded as the threshold, minimum value for the two estimators. Only one performance measure, f1-score will be used with the two estimators in a datasets which has been randomly divided. Despite having many classification algorithms, decision tree will be used for setting the dataset threshold. The next section provides experimental results after experimentation.

5.2.1 Abalone dataset

The first experiment has been performed using the Abalone dataset from the UCI machine learning repository. The accuracy threshold between the two values has been calculated and 0.5979 was obtained. Table 10 represent the performance values for the two performance estimators together with their differences.

Dataset	F1-Score		Difference (f1-score)
Sample Size	10 fold CV	Leave-one-out	
4177	0.7448	0.7452	-0.0004
2006	0.7502	0.7502	0
1000	0.7307	0.7334	-0.0027
750	0.6442	0.6473	-0.0031
500	0.6603	0.6599	0.0004
250	0.6514	0.608	0.0434
100	0.6216	0.5941	0.0275
50	0.5455	0.4878	0.0577

Table 10: Results for the 10 fold CV and leave-one-out estimation for the Abalone dataset

From table 10, the results can be evaluated using the f1-score difference between k-fold cross validation and leave-one-out method shown in the last column. For the whole dataset with 4177 instances, the difference between the two is -0.0004 f1. With 2006 instances there is a 0 f1-score difference between the two. Sample size with 1000 and 750 instances there is -0.0027 and -0.0031 differences in case of f1-score for the two estimation methods respectively.

However, for the sample size with 500 and 250 instances, f1-score difference between two performance estimators increases abruptly to 0.0004 and 0.0434 respectively. For the sample size with 100 instances the difference is 0.0275 and for the 50 instances the f1-score difference is 0.0577. Figure 25 represent the line graph for the dataset threshold experiment using the decision tree algorithm.

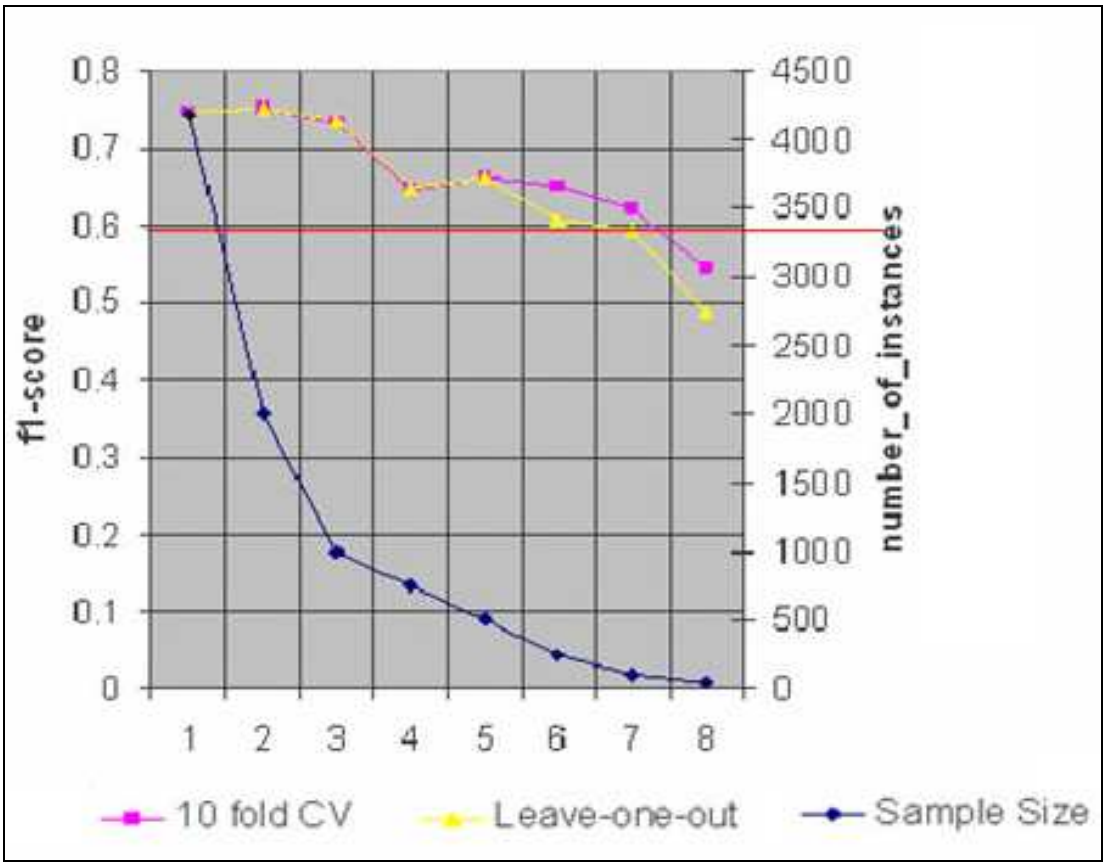

Figure 24: Line graph for the comparison of 10-fold cross validation and leave-fold cross validation for the Abalone dataset

From figure 24, looking at the accuracy threshold shown with the horizontal line, 10 fold cross validation still performs well while leave one out cross validation falls abruptly. This shows that for the dataset with 4177 the best performance estimator is 10 fold cross validation.

5.2.2 Contraceptive method choice

The second experiment for the dataset threshold has involved 1473 instances and 10 attributes. The accuracy for the f-1 score which is the minimum value for the performance of the estimators is 0.9966. This value will be the cutting point for the selection of the performance estimator.

The original Contraceptive method choice has 1473 instances. More description of the dataset has been presented in appendix A.

Dataset	F1-Score		Difference (f1-score)
Sample Size	10 fold CV	Leave-one-out	
1473	0.9983	0.9984	-0.0001
735	0.9979	0.9980	-0.0001
350	0.9986	0.9986	0
175	0.9970	0.9971	-0.0001
85	0.9933	0.9941	-0.008
40	0.9857	0.9873	-0.0016
20	0.9667	0.9744	-0.0077

Table 11: Results for the 10 fold CV and leave-one-out estimation for the Contraceptive method choice

As shown in table 11, the two performance estimators have very minor f1-score differences which range from 0 to -0.008. Using the human eye, the difference will not predict anything and also will not help predict the correct estimator. With 1473, 735 and 175 instances, the f1-score difference between the two performance estimators are -0.0001. With 350 instances there is no difference between the two while 85 instances the difference is -0.008. With 40 and 20 instances the difference is -0.0016 and -0.0077 respectively. The results presented in table 11 have been depicted in figure 25.

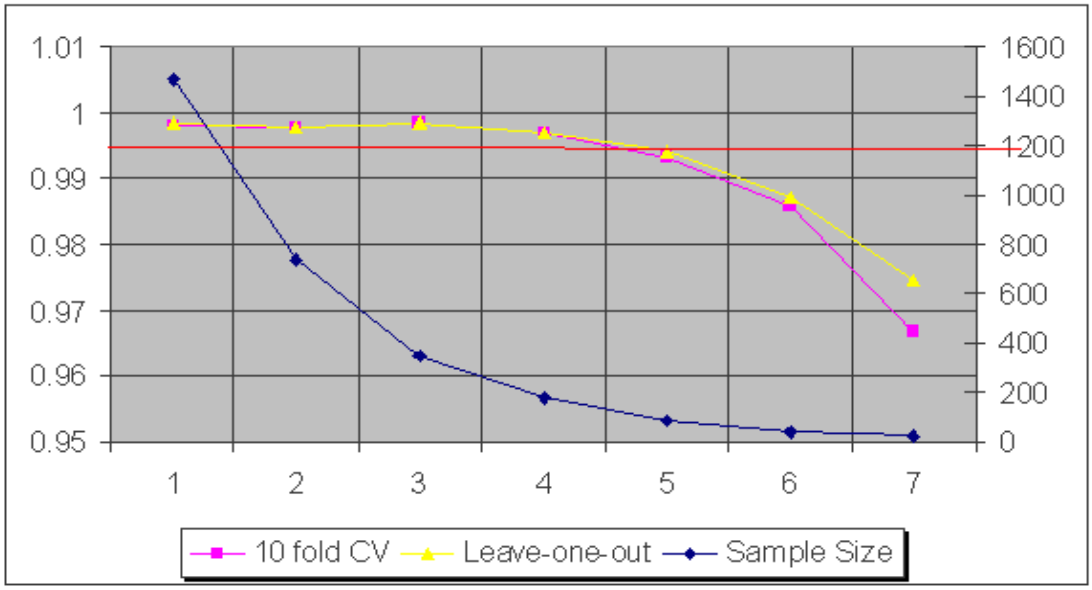

Figure 25: Line graph for the comparison of 10-fold cross validation and leave-fold cross validation for the Abalone dataset

From figure 25, at the accuracy threshold leave one out performs well while k-fold cross validation falls abruptly. Therefore, for 1473 instances leave one out method is the perfect performance estimator.

From the previous experiences with 4177 and 1473 the performance estimators are k-fold cross validation and leave-one-out method respectively. Between 4177 and 1473 there is a need for the experiments using 3000 and 2000. These two values will probably provide the separation of the use of the performance estimation methods.

5.2.3 Ozone Level Detection Dataset

For this section dataset with around 2000 instances will be used for estimating the threshold of the dataset as from the previous sections 4000 and more than 1000 instances have been used. This dataset contain 2536 instances and 73 attributes. The accuracy threshold for this dataset is 0.7856.

Dataset	F1-Score		Difference (f1-score)
Sample Size	10 fold CV	Leave-one-out	
2536	0.8799	0.8800	-0.0001
1268	0.8804	0.8805	-0.0001
634	0.8858	0.8858	0
317	0.8759	0.8759	0
158	0.8911	0.8912	-0.000
79	0.8101	0.8120	-0.0019
40	0.8190	0.8235	-0.0045
20	0.8000	0.8235	-0.0235

Table 12: Results for the 10 fold CV and leave-one-out estimation for the Ozone layer detection

From table 12, two datasets have no f1-score differences; the dataset with 634 instances and 317 respectively. With 2536 and 1268 instances the difference is -0.0001. Datasets with 79, 40 and 20 instances have -0.0019, -0.0045 and -0.0235 f1-score respectively. The values in table 12 have been presented in figure 26 with accuracy threshold presented as the horizontal line.

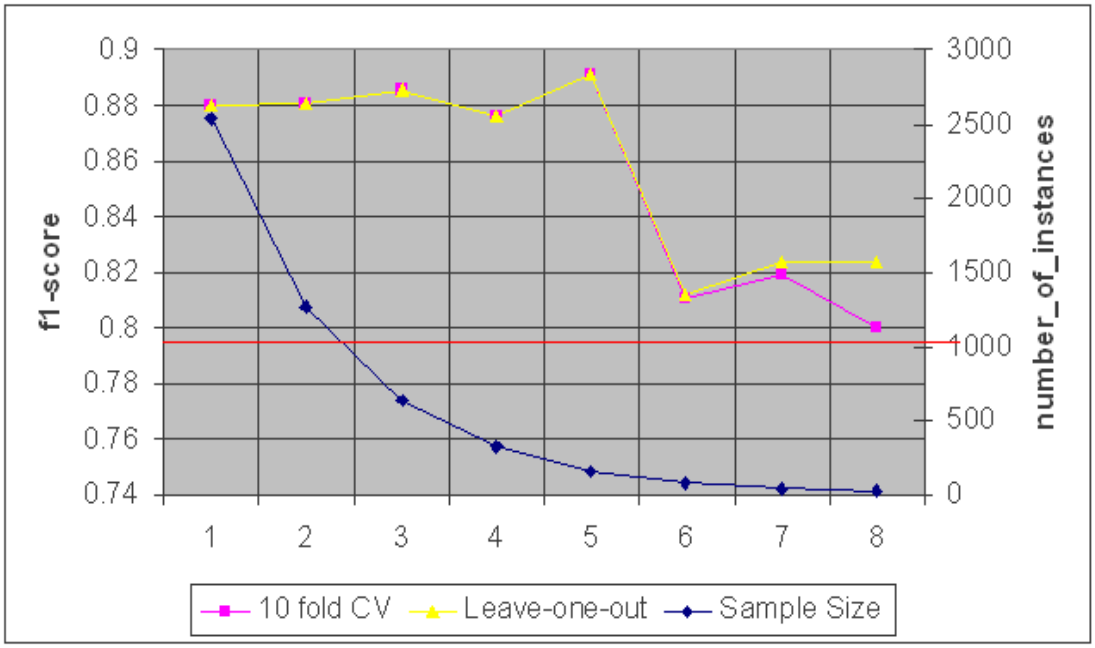

Figure 26: Line graph for the comparison of 10-fold cross validation and leave-fold cross validation for the Abalone dataset

As described as the beginning of this section, the experiment involves the dataset with more than 2000 instances. Using the accuracy threshold, it will be easy to notice the behaviour of the performance of the two estimators. When f1-score between the two values is nearby 0.82, estimators misbehave from one another as leave one out method continue to behave well, k-fold cross validation falls. Therefore, it can be concluded that, for the 2000 instances, leave-one-method is the perfect estimator.

5.2.4 Internet advertisement

This is the last experiment which will determine the dataset threshold for the two performance estimators. From the previous sections, experiments have been performed for the dataset with 4177, 2536 and 1473 instances and the performance estimators obtained are k-fold cross validation for the first dataset while the other two, leave-one-out cv has been identified as the performance estimator. This experiment lies between the obtained results.

This dataset contains 3278 instances and 1558 attributes. The dataset has been created to represent set of possible advertisement on the internet pages. More description of the dataset has been presented in Appendix A. The results of the experiment using this dataset have been plotted in table 13.

Dataset	F1-Score		Difference (f1-score)
Sample Size	10 fold CV	Leave-one-out	
3279	0.9902	0.9902	0
1639	0.9988	0.9988	0
819	0.9988	0.9988	0
409	0.9975	0.9975	0
204	0.9974	0.9974	0
102	1.000	1.000	0
51	1.000	1.000	0
25	1.000	1.000	0

Table 13: Result for the 10 folds cross validation and leave-one-out for the internet advertisement dataset.

There is no any difference between the two performance estimation methods. The results shown in table 13 are presented in a line graph in figure 27

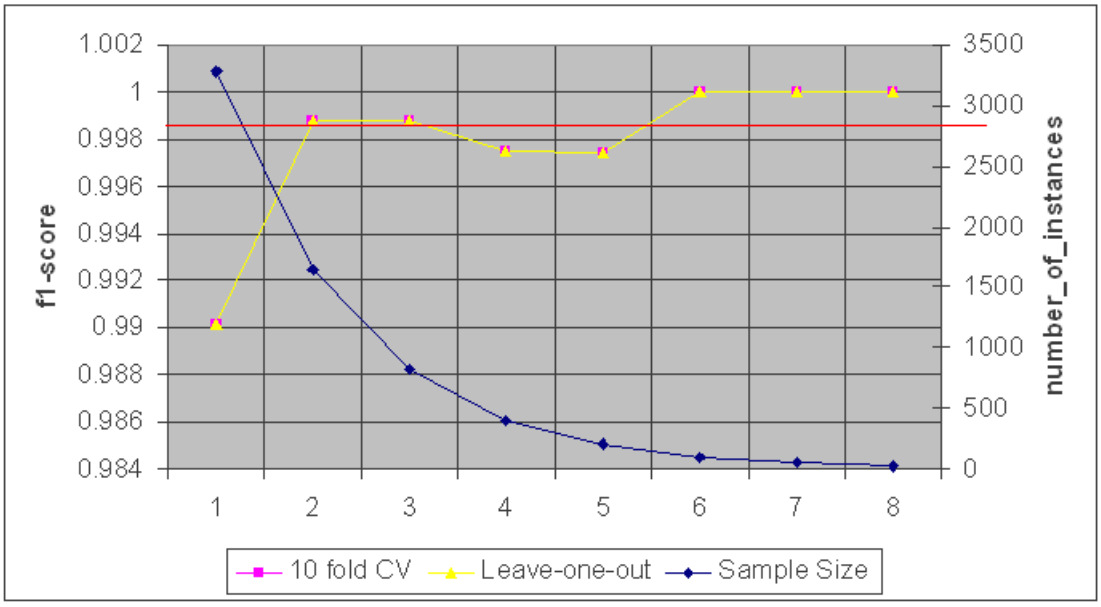

Figure 27: Line graph for the comparison of 10-fold cross validation and leave-fold cross validation for the Internet advertisement dataset

With the internet advertisement dataset, there is no dataset threshold as the two estimators flow together. Next section provides analysis of the results obtained in section 5.2.

5.3 Dataset threshold experimental result

From the experiments carried out in the previous section, dataset needs to be established. The result of this section will be incorporated into the proposed classifier workflow provided into the chapter 4.The overall results are presented in table 14

The main aim of this experiment was to establish number of instances which can result into the classification of the dataset as either small or medium. For the large datasets, holdout method as discussed in section is an appropriate method.

Sample size (number of instances)	Performance estimation method
4177	k-fold cross validation
3279	neutral
2536	Leave one out cv
1473	Leave one out cv

Table 14: number of instances versus performance estimation method

From table 14, with 4177 instances k-fold cross validation outweighs leave one out method and this means for this number of instances, k-fold is the appropriate method. With 2536 and 1473 instances, both supports leave one out method. The needed threshold is obtained when the number of instances is 3279. Therefore for the classifier workflow, the dataset threshold is 3279.

The obtained dataset threshold is closely related to the selection of the performance estimation method. Figure 28 represents dataset threshold and performance estimators for the classifier workflow.

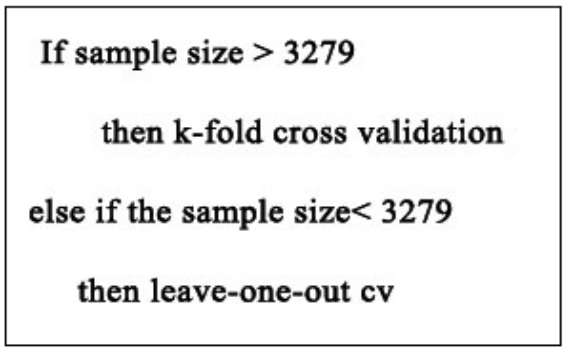

Figure 28: Dataset threshold for the classifier workflow

5.4 Key parameter settings

The aim of this section is to establish key parameters for each classification technique provided in table 7. The experiment for finding key parameters involves the use of one or two key parameters in a related algorithm where f1-score will be used as the performance measure and 10-fold cross validation as the performance estimation method. The results for each experiment will be provided in each section. For the parameters that do not show any changes in f1-score will be regarded as non-key parameters for such an algorithm.

5.4.1 Decision tree

In order to determine if the change in algorithm parameters values provides different performance measure values, decision tree is the first algorithm to be tested. With the decision tree, two attributes will be tested and the results will be related to performance measure (f1-score); maximal depth or maximum tree depth and minimal leaf size.

Maximum tree depth "*is a limit to stop further splitting of nodes when the specified tree depth has been reached during the building of the initial decision tree"* (IBM n.d.).

The change parameter values will imply maximal depth is the key parameter and experimenter has to test three different: low, medium and high. As shown in table 15, the three maximal depth values provides different f1-score results.

Maximal depth	F1-score
3	0.7151
10	0.6331
Number of features (16)	0.6185

Table 15: Maximal tree depth versus the f1-score for the decision tree

The second parameter to be tested for the decision tree is the minimal leaf size. Table 16 represents the values of the minimal leaf size plotted against f1-score.

Minimal leaf size	F1-score
3	0.7215
10	0.7230
Number of features (16)	0.7272

Table 16: Minimal leaf size against f1-score

5.4.2 Neural Network

Neural networks have a range of parameters such as number of hidden layers, number of hidden layer size which needs to be tested. Initially, the number of features will be tested to determine if there is any relationship between number of features, number of hidden units and performance measure. As from the previous subsection, three different values will be used in relation to the number of parameters determined as low, medium and high. These values; low, medium and high are related to the number of features in the dataset. Consider for example, if the dataset has 16 features, then this value should be related to low, medium and high value

With this experiment, the number of features is regarded as the medium value for the experiment. When the number of features is divided by two are related to the low value and when multiplied by two this becomes the high value. Table 11 represent the value of the number of hidden layers in relation to the f1-score.

Number of hidden layers	F1-Score
$\frac{Number\ of\ \ features}{2}$	0.7350
$Number\ of\ \ features$	0.7350
$Number\ of\ \ features * 2$	0.7350

Table 17: Number of hidden units and f1-score values for the neural network

With the neural network using number of hidden units, there is no difference in terms of the f1-score values. This implies that, the parameter is a non-key and will be eliminated in the last key parameters table.

The second parameter to be tested for the neural network is the hidden layer size. The results of this experiment have been shown in table 18. Three parameters values; low, medium and high are related to the number of features as in the previous experiment with the number of hidden layers.

Hidden layer size	F1-Score
$\frac{Number\ of\ \ features}{2}$	0.7308
$Number\ of\ \ features$	0.7324
$Number\ of\ \ features * 2$	0.7312

Table 18: Hidden layer size and the f1-score values for the neural network.

The results depicted in table 18 show changes in the f1-score values in relation to the change in number of features. For the classifier workflow, if an experimenter intends to use neural net then this is the parameter to deal with.

5.4.3 K-Nearest Neighbour

As discussed in chapter 2, the K-nearest neighbour works by calculating the distance of nearby instances. For better generalization, more than one nearest neighbour distance has to be calculated, that's why they are called k-nearest neighbour, where as *K* stands for the number of neighbours. For this dissertation, only k was measured and its results were related to the f1-score.The results obtained may indicate if the changes in parameter values have any effect to the performance measure.

For this experiment, the same three different values in relation to the parameter selected are tested in order to observe if there are any changes that happen when k-values are iterated. The k-values have been randomly selected for the experiment.

Value of *K*	F1-Score
3	0.6437
9	0.6811
16	0.6562

Table 19: *K*-values with f1-score for the KNN

Three *k* values were calculated, 3 was regarded as the lowest value, 9 as the medium value and the number of features as the maximum value. The results can be generalised as, change in k-values results into different f1-score value.

5.4.4 SVM

This technique is useful for data classification. Most of the users who are familiar with machine learning consider SVM to be simple in use but for non-familiar users it is easy to get unsatisfactorily results. There are several parameters which are set in relation to a kernel type and SVM type. With the classification techniques there are two SVM types; c-svc and nu-svc and four kernel types; linear, rbf, polynomial and sigmoid. The *c-svc* is a multi-class support vector machine for classification while *nu-svc* is a parameter with values ranging from 0 to 1. These two SVM types are the

same but differ in coverage of the parameters. The c-svc ranges from 0 to infinity while nu-svc has only two values 0 and 1.

For this dissertation LibSVM has been used and the gamma and cost parameter has been used to test the change in performance of the technique. The performance of the LibSVM while changing the values of the gamma parameter has been provided in table 20.

(γ)gamma parameter	F1-Score
3	0.5721
8	0.6594
16	0.6655

Table 20: γ parameter with f1-score for SVM

The second parameter is cost presented as *c*. From the results shown in table 21, this parameter does not indicate any changes to the f1-score values for different c parameter values. When the cost parameter equals to 3 the f1-score is 0.7456.The same value of 0.7456 is obtained when c equals to 8 and 16.Therefore, this indicates that cost is not a key parameter for the LibSVM.

(c)cost parameter	F1-Score
3	0.7456
8	0.7456
16	0.7456

Table 21: c parameter with f1-score

5.4.5 Random Forest

Two key parameters for the random forest are selected to test the change in the performance measure; *number of trees* that are grown and *maximum tree depth*. For

this experiment to work effectively and efficiently, three different values were selected in relation to each of the performance measure.

For the effects of the number of trees, maximum depth was set up to its default value of 10. Number of trees will be changed randomly so as to notice if there is any changes for the same dataset. If both parameters provides changes to the performance measure(s) used then both parameters need to be considered while performing experiments.

No_of_trees	F1-score
3	0.6699
10	0.6855
Number of features (16)	0.6969

Table 22: Number of trees and f1-score for the random forest

From table 22, there is a change in f1-score in relation to the number of trees developed. When the number of trees is low (3) the f1-score is 0.6699 and when the number of trees is medium, the f1-score increases. This change of the f1-score values indicates the parameter is a key for the performance of the algorithm.

The second parameter for testing the accuracy of the random forest is maximum depth. The maximum depth for the random forest was tested using three different values; the results will yield if changes in maximum depth will result into changes in the f1-measure.

Maximum depth	F1-score
3	0.6947
10	0.6855
Number of features(16)	0.7137

Table 23: Maximum depth and f1-measure for the random forest

Maximum depth provides the changes in the performance measures. This means that in random forest maximum depth is among the two key parameters which need experimenters' attention while comparing the performance of the classifiers or calculating the performance measure for each algorithm.

5.5 Parameter settings results

From the results of the experiments presented in section 5.4 where the main aim was to establish key and non-key parameters for the algorithms presented in section 2.5. The established parameters will allow the experimenter while using classifier workflow not to test as many values as possible, rather use the parameters proposed in table 24 while doing experiments. The overall results of the experiments have been presented in table 24.

With the decision tree, two parameters have been tested maximum depth and minimal leaf size. Both parameters tend to provide different f1-score values. Therefore, both are the key parameters for the decision tree algorithm.

The second algorithm is neural net where number of hidden layers and number of hidden layer size have been tested against the change in f1-score. The f1-score for the number of hidden layers do not change while values for the number of hidden layer size change. Therefore, number of hidden layers is non-key while number of hidden layer size is the key parameter for the neural network.
KNN as discussed in section 2.5.3 classify the newly fired instances by calculating distance to its nearest neighbours. With this algorithm only one key parameter can be tested against the f1-score; value of k which is the number of the neighbours.

SVM has been tested using two parameters gamma and cost parameters. The f1-score values are stagnant while gamma values are changing. From these results, gamma will be considered as the key parameter while cost is the non-key parameter.

Random forests have been tested using number of trees and the maximum depth. The f1-score changes with different parameter values. Therefore, both parameters are considered as the key parameters for the random forests algorithms.

Algorithm	Parameter					
	Parameter 1			Parameter 2		
Decision Tree	Low	Medium	High	Low	Medium	High
	Maximum depth			*Minimal leaf size*		
	3	10	NoF	3	10	NoF
	0.7151	0.6331	0.6185	0.7215	0.7230	0.7272
Neural Network	*Number of hidden layers*			*Number of hidden layer size*		
	NoF/2	NoF	NoF*2	NoF/2	NoF	NoF*2
	0.7350	0.7350	0.7350	0.7308	0.7324	0.7312
KNN	*Value of k*					
	3	9	16			
	0.6437	0.6811	0.6562			
SVM	*γ (gamma) parameter*			*C (Cost) parameter*		
	3	8	16	3	8	16
	0.5721	0.6594	0.6655	0.7456	0.7456	0.7456
Random Forest	*Number of trees*			*Maximum depth*		
	3	10	16	3	10	16
	0.6699	0.6855	0.6969	0.6947	0.6855	0.7137

Table 24: Learning algorithms parameter testing results

Table 25 presents the proposed key parameters for the five learning algorithms discussed in section 2.5; decision tree, neural net, KNN, SVM and random forests.

Algorithm	*Parameter 1*	*Parameter 2*
Decision tree	Maximum depth	Minimal leaf size
Neural network	Number of hidden layer size	
KNN	Value of k	
Support Vector Machine	Gamma parameter	
Random Forest	Number of trees	Maximum depth

Table 25: Algorithms key parameters

5.6 Conclusion

The chapter aimed at performing experiments for establishing benchmark settings from the proposed classifier workflow phases. The chapter started with the experiment for setting the dataset threshold where its results help in selection of the performance estimation method. The result of the dataset threshold experiment has been presented in section 5.3 where the threshold obtained was 3279 instances; this means when performing experiment, if the number of instances is 3279 or less then leave-one-out is the appropriate measure. On the other hand, if the number of instances exceeds 3279 then k-fold cross validation is appropriate.

The experiment for establishing key parameter settings for each classification technique has been presented in section 5.5 of this chapter. The results for these experiments have been analysed in section 5.5. These experiments have been performed to help the user of the classifier workflow to use the suggested parameters if an experiment involves any of the presented classification techniques in section 2.5.

6 Evaluation and Final Classifier Workflow

6.1 Introduction

This chapter is aimed at combining the proposed classifier workflow in chapter 4 and the results of the experiments performed in chapter 5 and the results of the evaluation so that the complete classifier workflow will be developed.

The chapter starts with the evaluation of the classifier workflow using the results obtained from the evaluation survey conducted to expert and non-expert users of machine learning and data mining. The classifier workflow has been presented in the second section of the chapter. The third and last section provides the walkthrough of using the classifier workflow.

6.2 Evaluation of the classifier workflow

The aim of this section is to analyse results and to provide recommendations obtained from the evaluation of the classifier workflow. A questionnaire was adopted as the evaluation methodology. Section 6.2.1 describes the participants and methodology adopted for the survey followed by survey results analysis for each question in the evaluation questionnaire in section 6.2.2. Recommendations from the respondents are provided in section 6.2.3.

6.2.1 Audiences and methodology

Since the survey aimed at understanding the applicability of the proposed classifier workflow to small scale classification projects, expert and non-expert users in machine learning and data mining were involved. Experts were involved as they have broad knowledge in machine learning and data mining so as they can easily identify efficiencies and deficiencies in the proposed workflow. On the other hand, non-experts were involved so as to understand their views based on the workflow. The evaluation involved 12 respondents where 6 were experts and 6 non-experts who are familiar with machine learning and data mining. Although the number of respondents seems to be poor, for evaluation only their input to the classifier workflow was needed.

6.2.2 Survey results analysis

From the questionnaire distributed, many recommendations were obtained. This section describes the analysis process undertaken, presents the results and analyses them. The analysis is based upon each question in the questionnaire and the results obtained.

- ***Respondents familiarity***

From the analysis conducted based on question 1, most of the respondents were familiar with machine learning and data mining as 74% were familiar, 13% were very familiar. These respondents were useful not only because they were experts in the area but also they can easily identify the efficiencies and deficiencies of the workflow.

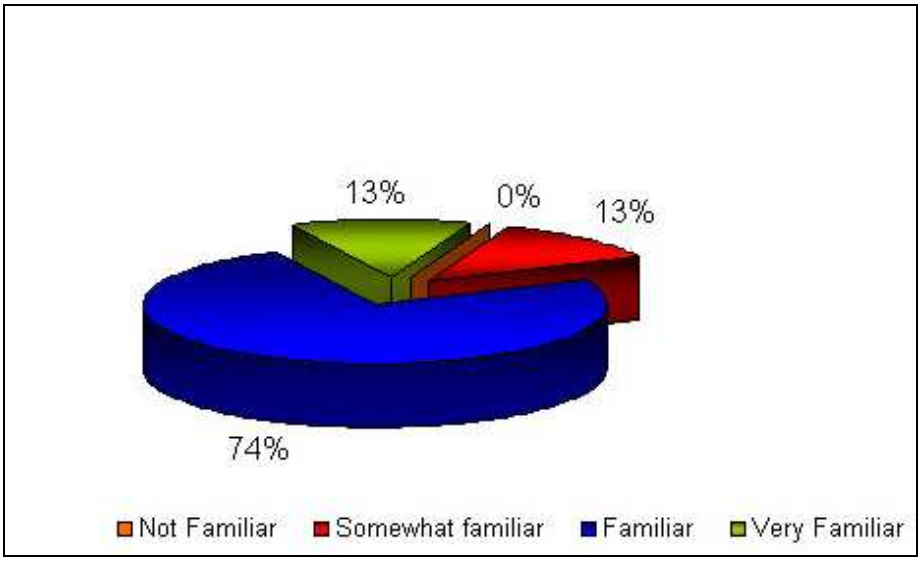

Figure 29: Distribution of the responses based on the familiarity to machine learning and data mining.

- ***Workflow understandability***

The analysis was done to determine the understandability of the workflow. 87% of the respondents agreed that the classifier workflow was understandable while 13% disagreed with the understandability of the workflow. Based on the analysis of question 1, where most of the respondents were familiar with machine learning and data mining, classifier workflow was considered understandable. An

interesting observation from this question was that, even non-experts agreed with the understandability of the workflow.

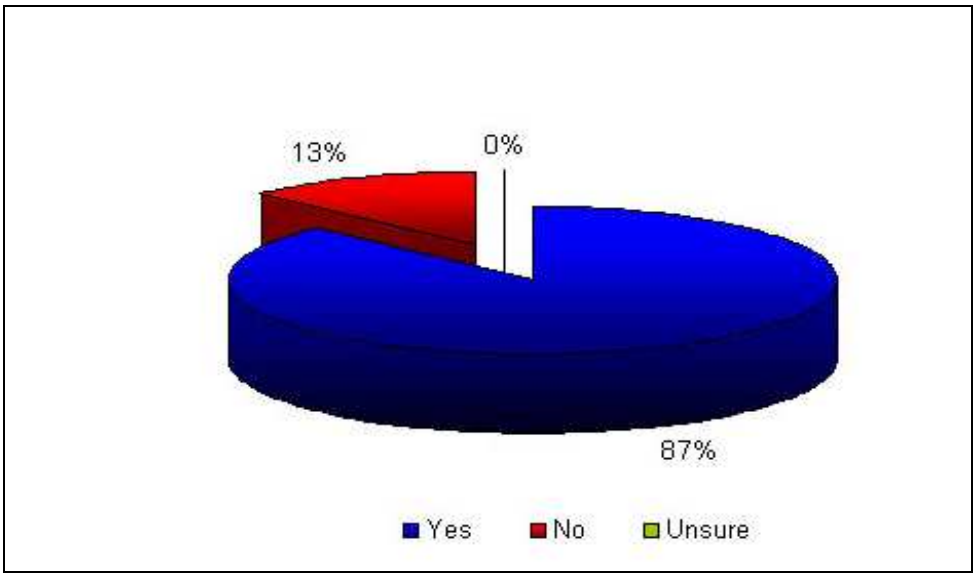

Figure 30: Distribution of the respondents based on the understandability of the workflow

- ***Number of phases***

The analysis was done using question 3 to determine if the classifier workflow contains enough phases for the comparison of classification technique in small scale projects. The results obtained showed 25% strongly agree while 49% agree leaving 13% each for those who do not know if the phases are enough and disagree. In total the number of respondents who think the workflow is understandable is 74%. This percentage comprise of machine learning experts which means, number of phases are enough. As evaluation involved non-experts, it was not possible for them to identify if the number of phases are enough for classification projects.

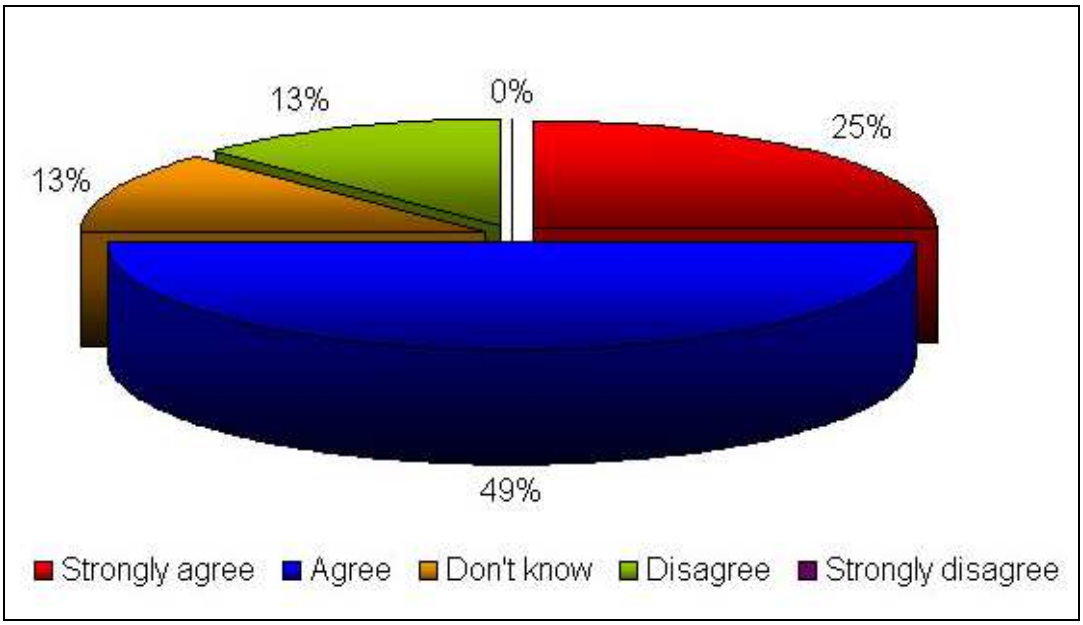

Figure 31: Distribution of the responses relating to the phases of the workflow

- ***Validity of the algorithm selection phase***

The analysis of question 4 shows that most of the respondents, 74%, agree on the second phase of the classifier workflow which suggests a way for evaluating algorithms that are applicable in such a domain. 13% of the respondents were not sure while 13% disagreed with the validity of the suggestions.

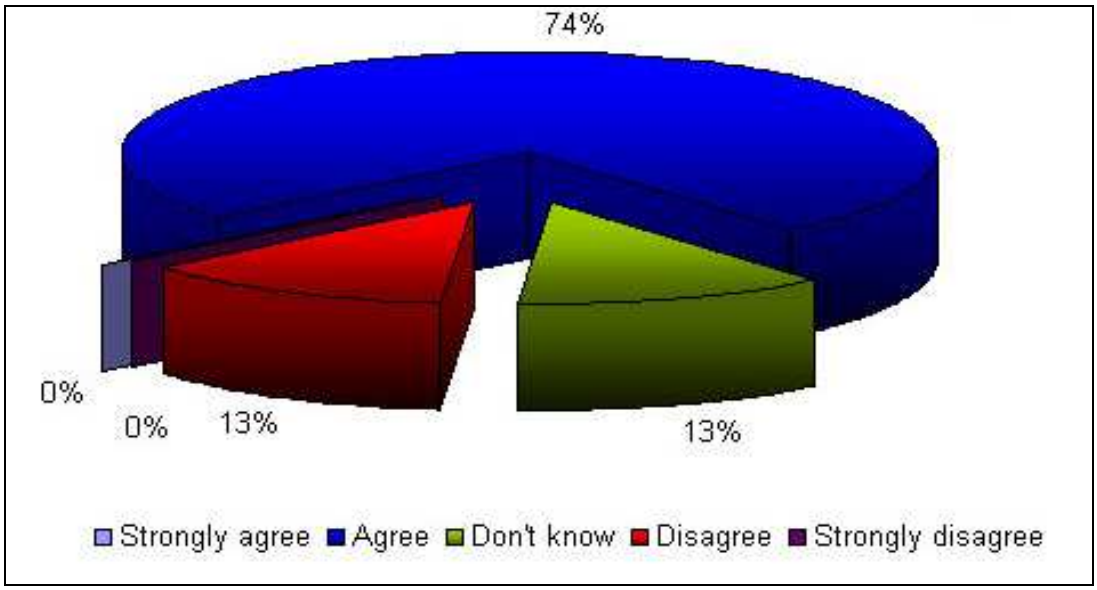

Figure 32: Distribution of the responses over the validity of the algorithm selection phase

- ***Validity of the dataset threshold***

The analysis in question 5 finds that 62% of the respondents agree that the dataset threshold is valid for the performance estimation methods. 25% were not sure as the threshold provided was valid or not leaving 13% disagreeing. From this question, respondents commented that the number of instances, 3279, for the dataset threshold were enough to categorise the dataset as either small or medium dataset.

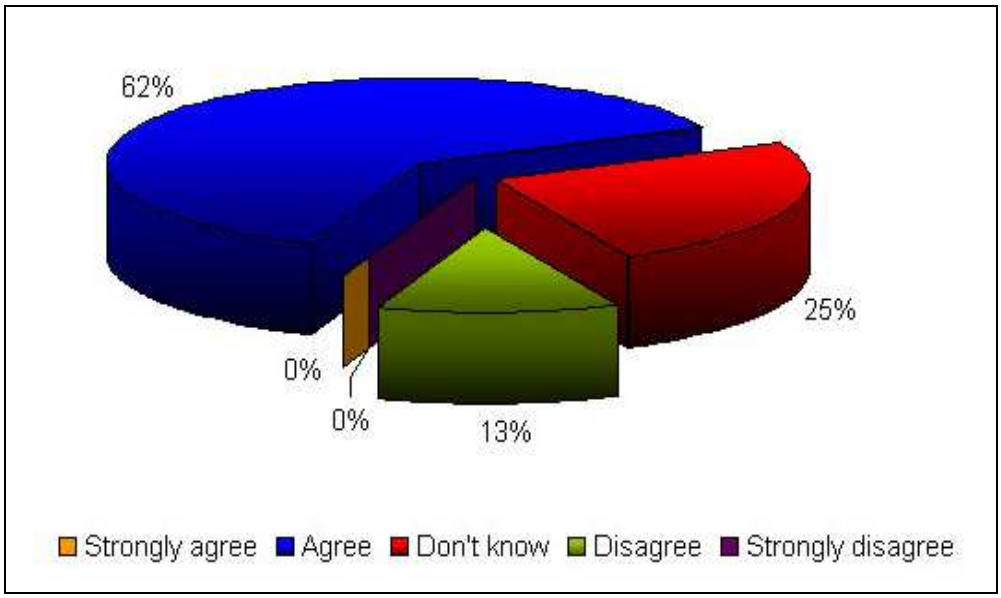

Figure 33: Distribution of the responses on the validity of the dataset threshold

- ***Key parameters settings***

The analysis of question 6 reveals some wonderful responses. Responses either agree or do not know. 75% of the respondents agree with the suggested way for setting key parameters leaving the remaining 25% without knowing if the suggested way is valid or not.

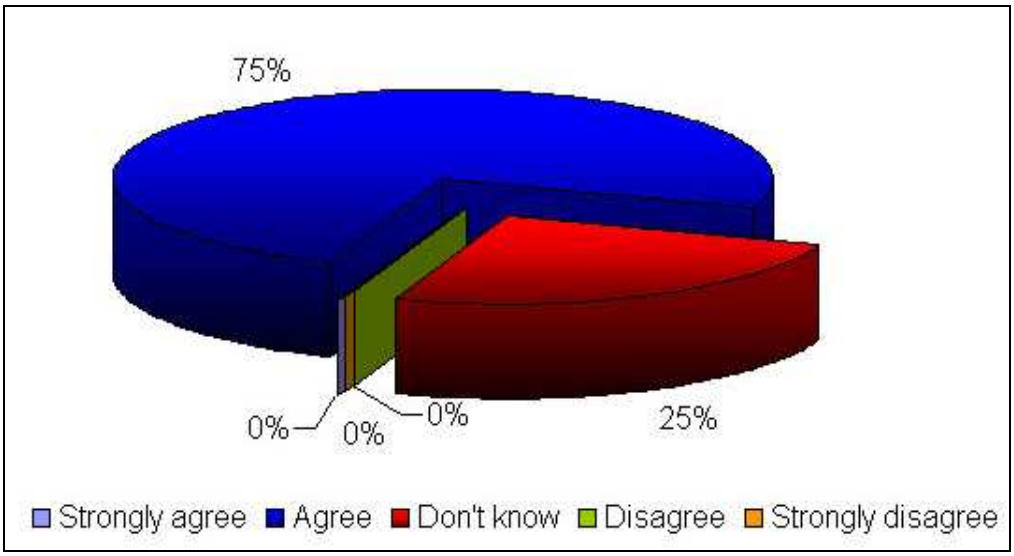

Figure 34: Distribution of the responses on key parameters settings

- ***Application to real world classification projects***

The analysis of question 7 shows that 62% of the respondents agree that the classifier workflow can be applied in real world classification projects, 13% strongly agree leaving 25% who do not know as whether the workflow should be applied to real world classification projects or not.

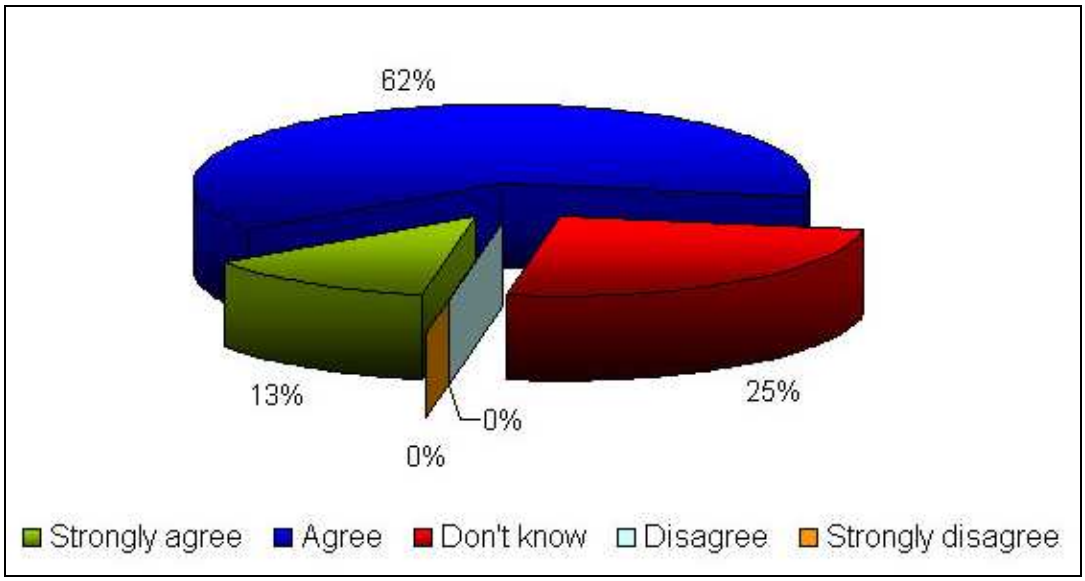

Figure 35: Distribution of the responses on the application to real world classification projects

6.2.3 Recommendations from respondents

The phases of the workflow however raised questions from the evaluators. The strengths and weaknesses of each phase outlined by the evaluators will be identified and provide the recommendations for the future work.

Preprocessing

As outlined in section 6.2 the classifier workflow did not emphasize on the third phase preprocessing as the author believes there are many ways for dealing with data problems such as substituting missing values with the mean, medium or maximum value. Evaluators of the workflow identified the preprocessing step as one of the limitations on using the workflow in real world problems. They suggest that more work needs to be performed in this phase. This limitation was applicable to expert and non-expert users.

Guidance

In case of non-expert users, that is, users who do not have specific machine learning experience then adequate guidance must be given in order to get through this phase. Respondents identified more guidance to be provided especially in the third phase and the fifth phase. Most concern was directed to non-experts as the identified phases require more machine learning and data mining knowledge.

Real world application

The process of applying the classifier workflow to the real world classification projects however highlighted some false alarm. From the analysis of question 7, there were 25% of respondents who did not know if the classifier workflow can be applied to real world classification projects and one recommendation was made by the expert relating to knowledge merging.

- **Knowledge merging:** The classifier workflow was developed for use in a small scale classification projects in a single application domain. There was no consideration on the process of merging the domain knowledge into the workflow. The evaluators of the workflow propose the addition phase between the first phase, experimental design and the second phase, preprocessing. The

task of this in-between phase is to transform the domain knowledge to machine understandable data and make it affect the process of the following steps.

From evaluation of the classifier workflow discussed in section 6.2 where efficiencies and deficiencies of the classifier work have been identified. Next section presents the final classifier workflow which is complete and comprehensive.

6.3 Final Classifier Workflow

The classifier workflow provided in this section is a continuation of the proposed classifier workflow provided in chapter 4. The developed workflow provides the same eight phases plus experimental results from the experiments performed in chapter 5. From the analysis of the second question where 87% of the respondents agree that the classifier workflow contains enough phases for machine learning and data mining classification projects. The phases of the workflow are outlined as follows

6.3.1 Phase I: Experimental design

This is the first phase in which objectives of performing the experiment are set out. This phase can also be refered as the plan for the experiment. Data exploration is also done in this phase whereby the user is expected to observe different behaviours of the dataset as having incomplete values, missing values or inconsistencies from the original dataset if applicable.

6.3.2 Phase II: Algorithm selection

In this phase, the experimenter is supposed to evaluate the algorithms depending on the objectives set in the first phase. The selection process involves the use of classification algorithm selection criterion table where algorithms are evaluated using different factors in relation to the objectives set. If the comparison of the classifiers is done for the bank where explanation (interpretability) is of interest then decision tree and KNN will be included and other algorithms will be eliminated. From this phase onwards the experimenter will use only algorithms that fit in a related application domain and hence reduce the rework rate.

Algorithm	Classification algorithm selection criterion		
	Interpretability	Training Speed	Testing Speed
Decision Tree	✓	✗	✓
Neural Network	✗	✗	✓
KNN	✓	✓	✗
SVM	✗	✗	✓
Random Forest (RF)	✗	✗	✓

Table 26: Classification algorithm selection criterion

6.3.3 Phase III: Preprocessing

Recommendations from the respondents of the evaluation survey suggest more guidelines to be provided for non-expert users while dealing with the third and fifth phase. This is the third phase where the explored data in phase one are considered and fixed. Missing and inconsistent values in the datasets are considered. In various machine learning and data mining software, data preprocessing are supported. For non-expert users, *median* is an appropriate value while dealing with missing values. For the experts there is nothing to worry as they have the ability of using more than the suggested median.

6.3.4 Phase IV: Performance estimation method selection

In machine learning there are three categories of performance estimation methods which are used differently according to the amount of instances in a dataset. Hold out method is used when dataset has enough examples to be called large dataset while k-fold is for small to medium datasets and leave-one-out method for small datasets. There is a rare possibility of having large enough dataset and hence only two estimators will be considered for the classifier workflow.

From the experiments performed in chapter 5, the threshold of the dataset has been identified. With this phase, an experimenter is expected to select one method from

two available performance estimation methods for small to medium datasets. This phase is mostly related to the first phase when data in the dataset are explored. Experimenter has to use the number of instances in the dataset to select appropriate performance estimation method. Figure 38 provides *if-then* rules between the number of instances (sample size) and the two performance estimation methods; k-fold cross validation and the leave-one-out method. Holdout method will only be used if dataset contain enough many instances.

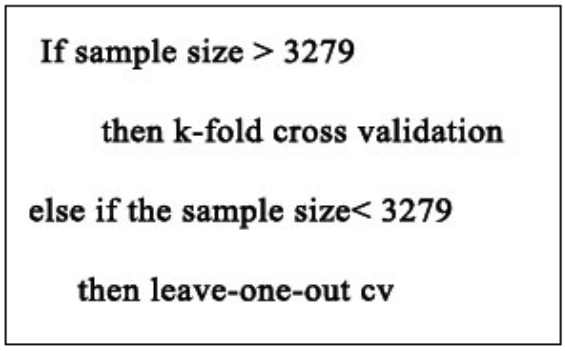

Figure 36: *if-then* rules for the selection of the performance estimation method in the classifier workflow

6.3.5 Phase V: Performance Measures

Different performance measures can be used for different datasets. Based on class distributions, datasets can be categorized as either balanced or unbalanced. The dataset is considered unbalanced if the classes are not approximately equally represented while balanced dataset contains equal mix of negative and positive examples. For balanced datasets, an experimenter can use predictive accuracy, error rate or ROC analysis.

With unbalanced dataset a number of performance measures exist namely precision, recall and f1-score. The experimenter is expected to use only f1-score and leave out precision and recall due to the biasness of the two measures in relation to each other as in the same experiment an experimenter can get higher precision and low recall which will then be hard to evaluate and vice versa. To avoid this biasness, f1-score which is a product of precision and recall will be used in classifier workflow while comparing classifiers in unbalanced datasets.

6.3.6 Phase VI: Algorithm parameters

This is the sixth and last phase that makes up the inner circle of the classifier workflow. If the experimenter is using any classification algorithm discussed in section 2.5, table 26 has to be used for the selection of the parameters. If the classification algorithm is not in the table then an experimenter is supposed to test parameter using three different values set as low, medium and high.

Algorithm	*Parameter 1*	*Parameter 2*
Decision tree	Maximum depth	Minimal leaf size
Neural network	Number of hidden layer size	
KNN	Value of k	
Support Vector Machine	Gamma parameter	
Random Forest	Number of trees	Maximum depth

Table 27: Classification algorithm parameters

There are more than five classification algorithms that are shown in table 26. For those algorithms which are not presented in table 26, experimenter is supposed to use table 27 where learning algorithms are plotted against parameters with three values named as low, medium and high.

Algorithm	Parameter					
	Parameter 1			Parameter 2		
	Low	Medium	High	Low	Medium	High
Algorithm 1						
Algorithm 2						

Table 28: Algorithm parameters settings table

6.3.7 Phase VII: Experimentation

This is the seventh phase where the experiments are performed after complete settings of the classifier workflow. Experiments forms the first phase of the outer circle of the classifier workflow. The results will be recorded until the eighth phase where the evaluation will be performed.

6.3.8 Phase VIII: Evaluation

The results of the experiments are evaluated and the conclusion as to which algorithm is better than the other for such an application domain is identified. With this phase, experimenter will be able to make conclusion with confidence as which algorithm is appropriate for one or more application domain in relation to the objective of the experiment set. Statistical tests, such as Mc Nemar and 5 x 2 cross validated paired *t* test will be used to test the significance of the differences between the results as either the difference is significant or just due to chance.

Performance Measures	Algorithm	
	Algorithm A	Algorithm B
Accuracy		
Precision		
Recall		
F1-Score		
Statistical tests (McNemar's Results)		

Table 29: Algorithm performance evaluation table

6.4 Walkthrough of the classifier workflow

Due to the scope of this dissertation, it was not possible to test the classifier workflow on a real world classification project. This section tests the application of the classifier workflow on the hypothetical case study. The case study chosen will reflect and encounter all obstacles and incorporate them in a walkthrough. This section walks closely with previous section.

6.4.1 Given situation of the walkthrough

The walkthrough is based on the credit approval dataset which contains confidential data from a financial institution for credit card applications and is provided in the UCI repository. The dataset is made up of 690 instances, each with 15 normal attributes plus one class attribute. These attributes are -continuous, nominal attribute with small values and nominal attributes with large values. The dataset also contains some missing values. Further description of the dataset has been provided in appendix A.

6.4.2 Problem statement

The overall goal is to improve the classification of the credit card applications. This can be achieved through development of a model using classification techniques. Applicants reserve the right to know the outcome of their applications as when the application is granted or not.

6.4.3 Walkthrough

The walkthrough will define all the phases that need to be passed. For the classifier workflow eight phases will be used and their descriptions have been provided in the previous section. The phases are as follows:

- Experimental design
- Algorithm selection
- Preprocessing
- Performance estimation method
- Performance measure
- Algorithm parameters

- Experimentation
- Evaluation

1. Experimental design

This is the initial phase for machine learning and data mining classification projects where the objective(s) of the project are identified and the data is explored.

Project objective:

To develop a model for the classification of credit applications and the model is required to provide explanation.

After analysing the dataset the following information were obtained

Walkthrough

i. Dataset has 690 instances
ii. 15 normal attributes and 1 class attribute
iii. Dataset contain missing values
iv. Classes are equal distributed: 307 were "+" instances while "–"instances were 387

Information discovered in this phase is important and will be refered to different phases of the project. From this initial phase, according the classifier workflow, the second phase is algorithm selection.

2. Algorithm selection

In the second phase algorithms need to be evaluated depending on the problem statement. The classification algorithm selection table can be used to evaluate the abilities and disabilities of the learning algorithms depending on the problem statement.

The evaluation of the learning algorithms has to consider the ability of the model on providing explanation. From classification algorithm selection table, the first column names interpretability will only be considered as in the objective of performing an experiment, it has been identified that applicants reserve the right to know the outcome of their applications therefore learning algorithms that provide explanation need to be considered.

Learning algorithms with cross symbols from this point onwards are left out as they do not fit the objective of the application domain.

Algorithm	Classification algorithm selection criterion		
	Interpretability	Training Speed	Testing Speed
Decision Tree	✓	✗	✓
Neural Network	✗	✗	✓
KNN	✓	✓	✗
SVM	✗	✗	✓
Random Forest (RF)	✗	✗	✓

Walkthrough

Learning algorithms that provides explanations

- Decision tree
- KNN

3. Preprocessing

This phase deals with correcting data problems before applying learning algorithms. For non-expert users it has been proposed to use the median while replenishing missing values. With the credit approval dataset 5% of all instances in the dataset has one or more missing values.

Walkthrough

After data exploration in phase 1, it has been noticed that

- Dataset contains missing values
- Classes are equally distributed

4. Performance estimation method

From three performance estimators discussed in section 3.5 and the dataset threshold obtained in chapter 5, the performance estimation method needs to be selected for the dataset with 690 instances

```
If sample size > 3279
      then k-fold cross validation
else if the sample size< 3279
    then leave-one-out cv
```

Walkthrough

From the dataset threshold, which is 3279 instances, credit approval dataset is below threshold and hence leave-one-out method will be used while calculating the performance of the classifier

After the selection of the performance estimation method, according to the classifier workflow, selection of the performance measure(s) follows.

5. Performance measures

The performance measure to be used in a related dataset is typically related to the distribution of the class attributes. If the class attributes are equally distributed then accuracy, error rate and ROC graphs may be used. With unbalanced datasets; precision, recall and f1-score might be used.

Walkthrough

From the analysis and exploration done in phase 1, the dataset is equally distributed and hence accuracy (error rate) or the ROC graph may be used.

6. Algorithm parameters

Before committing to experiments, parameters needs to be set for the two algorithms selected in the second phase of the classifier workflow. Decision tree and KNN are among the algorithms where their key parameter values have been proposed by the author.

Walkthrough

Decision trees have been identified to have two key parameters; namely maximum depth and minimal leaf size while KNN performance is affected by the different *k* values. These key parameters need to be tested using three parameter values: low, medium and high.

Algorithm	***Parameter 1***	***Parameter 2***
Decision tree	Maximum depth	Minimal leaf size
Neural network	Number of hidden layer size	
KNN	Value of k	
Support Vector Machine	Gamma parameter	
Random Forest	Number of trees	Maximum depth

This phase has identified key parameters for two learning algorithms selected in the second phase. After knowing key parameters of the learning algorithms then experimentation phase must begin while following an eight phase classifier workflow.

7. Experimentation

This phase involves combining all the setting from the first six phases. Information needed include learning algorithms, performance estimation method, performance measures and one or two key parameters for each learning algorithm

Walkthrough

Learning algorithms: decision trees, k-nearest neighbour

Performance estimation method: leave-one-out cross validation

Performance measure: accuracy
Key parameters: Decision tree-maximum depth
Minimal leaf size
K-nearest neighbours: value of k

The performance measure value is calculated for each learning algorithm.

8. Evaluation

This is the final phase where results from the previous experimentation phase are evaluated using statistical tests. As discussed in chapter 3, statistical tests are used to assess the significance of the differences between learning algorithms.

Performance Measures	Algorithm	
	Algorithm A	Algorithm B
Accuracy		
Precision		
Recall		
F1-Score		
Statistical tests (McNemar's Results)		

For the classifier workflow, evaluation has been identified as the last phase and the result is the classifier which fits the objective of the domain for performing the experiment.

This walkthrough serves as a hypothetical case study for the real classification project. It showed each phase of the workflow and how to link the information gained from different phases. The main aim for this walkthrough was to use dataset from any domain and to walk through the classifier workflow until the classification project was successfully completed.

6.5 Conclusion

This chapter was aimed at developing the complete and comprehensive classifier workflow for the use in small scale classification projects. The chapter was divided into three sections. The chapter was introduced before the discussion on the eight phases of the workflow.

The developed workflow was then given to the expert users in machine learning for evaluations. These results will indicate if the workflow requires some more changes or not and will serve as the future work in the conclusion.

7 Conclusions and future work

The chapter is aimed at summarising the findings of the dissertation and to draw a conclusion. The research is critically evaluated and the recommendations are made. Section 7.1 compares the results between the aims and objectives of the dissertation provided in chapter 1 with the outcome of the work done up to chapter 6. Section 7.2 provides conclusions drawn from the chapter 1 to chapter 6 while section 7.3 presents the future work. Summary of the chapter is provided in section 7.4.

7.1 Research evaluation

The research evaluation includes summary of the dissertation and a revision on the aim and objectives of this dissertation.

7.1.1 Summary of the dissertation

Chapter 1: The aim of this chapter was to introduce the dissertation and was divided into four sections. The chapter was introduced before an overview of the project in the second section. The main research problem was provided in the third section followed by the project aim and objectives in the last section.

Chapter 2: Provides broad discussion of machine learning. The main aim of the chapter was to provide the background materials for those who are unfamiliar with machine learning and its related concepts. Different application domains of machine learning were discussed followed by the discussion on two categories of machine learning; supervised and unsupervised learning. This dissertation was aimed for supervised machine learning and hence the previous section was for the purpose of showing the two related categories of machine learning. Later discussion was on classification techniques which are used for the creation of the classifiers in different domains. Existing workflows was reviewed and analysed and their strengths and weaknesses in machine learning and data mining was considered for the development of the classifier workflow.

Chapter 3: Classification evaluation methods were discussed in this third chapter of the dissertation. These methods were categorised into three, performance measures,

statistical tests and performance estimation methods. The discussion on six performance measures together with their evaluation was provided. The later discussion was based on the review of five statistical tests for measuring the significance of the differences between learning algorithms followed by the review on three performance estimators.

Chapter 4 provides the proposed iterative classifier workflow from secondary literature performed in chapter 2 and chapter 3. The proposed workflow comprised of eight phases which were divided into two circles, inner circle and outer circle. The inner circle was made up with experimental design which directly connects to the data in a dataset followed by algorithm selection, preprocessing, performance estimation methods selection and performance measures and algorithm parameters. The outer circle was made up of experimentation which was performed after algorithm parameters and evaluation as the last phase. There were unknown settings from different phases of the workflow which needed some more work. These unknown settings were provided in the last section of the chapter.

Chapter 5 was aimed at looking for the values for different unknown settings of the classifier workflow identified in chapter 4. Three different unknown settings experiments were performed setting the dataset threshold and the selection of the performance estimation method. The last experiment was for testing the effect in change of parameters in different classification techniques. The outcome of the last experiment was to provide a way for looking for key parameter value that would provide high performance when applied to its related classification technique.

Chapter 6 was aimed at applying the results of the experiments in chapter 5 to the proposed classifier workflow in chapter 4 so that a complete classifier workflow to be developed. Full and final complete classifier workflow was then evaluated by expert users to identify the applicability of the phases of the workflow to the small scale classification projects in machine learning. The evaluation was also done to identify the weaknesses of the workflow to real world applications.

Chapter 7 summarises the dissertation, draws conclusion and suggests future work from the research project.

7.1.2 Aims and objectives

The principal aim of this dissertation was to develop a classifier workflow for the comparison of the classification techniques in a single application domain. The challenge over machine learning and data mining projects is the lack of complete, efficient and effective workflow that provides guidelines to its non-expert users while performing experiments. The classifier workflow involved a number of phases which needed to be completed.

The following objectives have been achieved throughout the dissertation and contributed to the outcome obtained

1. To investigate and explore machine learning and techniques used while creating models for small scale projects.

2. To explore and evaluate a number of workflows and qualify their phases for small scale machine learning and data mining projects. The analysis of the existing workflows will provide inputs to the proposed classifier workflow.

3. To investigate classification evaluation schemes. The evaluation is categorised into three parts, performance measures, statistical tests and performance estimation methods. The performance measures such as accuracy, precision and recall are used to measure the performance of classification techniques while statistical tests are used for assessing the significance of the difference between learning algorithms. The performance estimation methods will be used to estimate the performance of the classifiers produced.

4. To perform experiments for setting up unknown settings of the proposed classifier workflow.

5. To produce complete and comprehensive classifier workflow which is industry neutral

6. Evaluate the classifier workflow using a closed-ended questionnaire distributed to expert and non-expert users in machine learning.

7.2 Conclusions of the dissertation

The author regards the following as the key recommendations drawn from the research carried out.

1. Machine learning and data mining classification projects involve complex comparative experiments which if not done correctly can result into experimenters' to make statistically invalid conclusion. From the research conducted in this dissertation there is a lot of work which have been done on both statistics and machine learning but little work has been done on the integration of the two fields for non-experts in statistics. As statistics provides important statistical tests that are used in machine learning to measure the significance of the differences between learning algorithms, it becomes hard for non-experts in the two fields to make use of the tests.

2. With machine learning and data mining projects, there is a need of having the workflow or sometimes refered as life cycles for the experimenters to follow as the guideline while developing classifiers or assessing the performance of the learning algorithms in certain domains. There is a number of existing workflows which have been developed for large scale projects. Empirical research has shown that there are few or no workflows which have been developed for the small scale classification projects in these two fields.

3. From the existing workflows which have been developed for machine learning and data mining projects, there is no coverage for non-expert users. The developers of the workflows which are mostly companies they only incorporate expert users. Classification projects can be performed even by non-experts as long as they are provided guideline. The research literature has identified the gap between machine learning and data mining existing workflows and the coverage of non-expert users. This has shown the requirement of the classifier workflow to cover the gap identified.

4. Classification techniques can be categorised as supervised or unsupervised based on the existence of the *"supervisor"*. This categorisation of the classification techniques has been identified useful particularly when attempting to look for techniques to incorporate into the experiments in such an application domain.

5. The research also outlines the need for different application domains to perform experiments and establish their own thresholds and algorithm parameters. For the experiments that need to be taken in specific domain, there is datasets with different characteristics; there is a need to establish the dataset threshold, performance estimation method that is going to be used and algorithm parameters for the classification technique(s) prior to experimentation.

6. The dissertation identified phases that need to be performed for a successful small scale classification projects. It can be concluded that the workflow must comprise the following
    - Experimental design or objective definition
    - Algorithm selection
    - Preprocessing
    - Performance estimation method selection
    - Performance measures
    - Algorithm parameters
    - Experimentation
    - Evaluation

7.3 Future research

This section lists a number of suggestions for further research in this area

Addition of phases

The current phases of the classifier workflow started from the experimental design to the evaluation. As outlined in the previous research literature over the existing

workflows in section 2.6, there is a need of incorporating more phases up to deployment. The lack of deployment phase may result into the same problems as outlined by Collier et al. (1998) over the traditional KDD process model. This will result into an experimenter to be able to perform the initial steps perfectly and hang out with the results after knowing which algorithm is better than the other in such an application domain.

Preprocessing phase

From the evaluation of the classifier workflow using questionnaire, experts in machine learning have identified the third phase, preprocessing, as a phase which results into limitations in using the classifier workflow. This phase which is huge and can be the research project itself has much effect on the performance of the learning algorithms. With the classifier workflow this phase was generalised as the author proposed that there is many ways for dealing with the data problems while in reality for non-expert users of the workflow this becomes hard without guidelines from expert users. For future work on the classifier workflow more work needs to be done on the third phase so that both expert and non-expert users to adopt the workflow in their applications without guidance.

Industrial testing

Due to time limit the workflow was evaluated by experts in machine learning using the closed-ended questionnaires. However for the workflow to be applicable in industries needs to be tested by two or more industries with different application domains. The author proposes more to be done in its evaluation so that to incorporate testing in different application domains.

7.4 Summary

Chapter 7 presented the overall conclusions of the research taken out in this dissertation and areas identified for future researchers. From the literature review, experiments and survey undertaken for the evaluation of the questionnaire, four main research areas are identified for future research.

Future research areas are also discussed in this chapter. Suggestions outlined in this chapter include addition of more phases so that the workflow would include up to deployment phase, more research on preprocessing phase which is a source of errors in machine learning and data mining classification projects, as the workflow is for a single application domain; industrial testing of the workflow is needed and more experiments on the dataset threshold in order to establish different dataset thresholds for different application domains.

Appendix A

Abalone dataset

Title of Database: Abalone data

2. **Sources:**

(a) **Original owners of database:**

Marine Resources Division
Marine Research Laboratories - Taroona
Department of Primary Industry and Fisheries, Tasmania
GPO Box 619F, Hobart, Tasmania 7001, Australia
(Contact: Warwick Nash +61 02 277277, wnash@dpi.tas.gov.au)

(b) **Donor of database:**

Sam Waugh (Sam.Waugh@cs.utas.edu.au)
Department of Computer Science, University of Tasmania
GPO Box 252C, Hobart, Tasmania 7001, Australia

(c) **Date received:** December 1995

3. **Past Usage:**

Sam Waugh (1995) "Extending and benchmarking Cascade-Correlation", PhD thesis, Computer Science Department, University of Tasmania.

Test set performance (final 1044 examples, first 3133 used for training):

24.86% Cascade-Correlation (no hidden nodes)
26.25% Cascade-Correlation (5 hidden nodes)
21.5% C4.5
0.0% Linear Discriminate Analysis
3.57% k=5 nearest neighbour (Problem encoded as a classification task)

4. **Relevant Information**

Predicting the age of abalone from physical measurements. The age of abalone is determined by cutting the shell through the cone, staining it, and counting the number of rings through a microscope -- a boring and time-consuming task. Other measurements, which are easier to obtain, are used to predict the age. Further information, such as weather patterns and location (hence food availability) may be required to solve the problem.

From the original data examples with missing values were removed (the majority having the predicted value missing), and the ranges of the continuous values have been scaled for use with an ANN (by dividing by 200).

Data comes from an original (non-machine-learning) study:
Warwick J Nash, Tracy L Sellers, Simon R Talbot, Andrew J Cawthorn and Wes B Ford (1994) "The Population Biology of Abalone (_Haliotis_species) in Tasmania. I. Blacklip Abalone (_H. rubra_) from the North Coast and Islands of Bass Strait", Sea Fisheries Division, Technical Report No. 48 (ISSN 1034-3288)

5. **Number of Instances:** 4177

6. **Number of Attributes**: 8

7. **Attribute information:**

Given is the attribute name, attribute type, the measurement unit and a brief description. The number of rings is the value to predict: either as a continuous value or as a classification problem.

Name	Data Type	Meas.	Description
Sex	nominal		M, F, and I (infant)
Length	continuous	mm	longest shell measurement
Diameter	continuous	mm	perpendicular to length
Height	continuous	mm	with meat in shell
Whole weight	continuous	grams	whole abalone
Shucked weight	continuous	grams	weight of meat
Viscera weight	continuous	grams	gut weight (after bleeding)
Shell weight	continuous	grams	after being dried
Rings	integer	+1.5	gives the age in years

Statistics for numeric domains:

Length	*Diam*	*Height*	*Whole*	*Shucked*	*Viscera*	*Shell*	*Rings*	
0.075	0.055	0.000	0.002	0.001	0.001	0.002	Min	1
0.815	0.650	1.130	2.826	1.488	0.760	1.005	Max	29
0.524	0.408	0.140	0.829	0.359	0.181	0.239	Mean	9.934
0.120	0.099	0.042	0.490	0.222	0.110	0.139	SD	3.224
0.557	0.575	0.557	0.540	0.421	0.504	0.628	Correl	1.0

8. **Missing Attribute Values**: None

9. **Class Distribution:**

Class	Examples
1	1
2	1
3	15
4	57
5	115
6	259

7	391
8	568
9	689
10	634
11	487
12	267
13	203
14	126
15	103
16	67
17	58
18	42
19	32
20	26
21	14
22	6
23	9
24	2
25	1
26	1
27	2
29	1
-----	----
Total	4177

Contraceptive Method Choice

Title of the Database: Contraceptive Method Choice

2. **Sources:**

(a) **Origin:** This dataset is a subset of the 1987 National Indonesia Contraceptive Prevalence Survey

(b) Creator: Tjen-Sien Lim (limt@stat.wisc.edu)

(c) Donor: Tjen-Sien Lim (limt@stat.wisc.edu)

(c) Date: June 7, 1997

3. **Past Usage:**

Lim, T.-S., Loh, W.-Y. & Shih, Y.-S. (1999). A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-three Old and New Classification Algorithms. Machine Learning. Forthcoming. (ftp://ftp.stat.wisc.edu/pub/loh/treeprogs/quest1.7/mach1317.pdf) or (http://www.stat.wisc.edu/~limt/mach1317.pdf)

4. **Relevant Information:**

This dataset is a subset of the 1987 National Indonesia Contraceptive Prevalence Survey. The samples are married women who were either not pregnant or do not know if they were at the time of interview. The problem is to predict the current contraceptive method choice (no use, long-term methods, or short-term methods) of a woman based on her demographic and socio-economic characteristics.

5. **Number of Instances:** 1473

6. **Number of Attributes**: 10 (including the class attribute)

7. **Attribute Information:**

1. Wife's age	(numerical)	
2. Wife's education	(categorical)	1=low, 2, 3, 4=high
3. Husband's education	(categorical)	1=low, 2, 3, 4=high
4. Number of children ever born	(numerical)	
5. Wife's religion	(binary)	0=Non-Islam, 1=Islam
6. Wife's now working?	(binary)	0=Yes, 1=No

7. Husband's occupation	(categorical)	1, 2, 3, 4
8. Standard-of-living index	(categorical)	1=low, 2, 3, 4=high
9. Media exposure	(binary)	0=Good, 1=Not good
10. Contraceptive method used	(class attribute)	1=No-use 2=Long-term 3=Short-term

7. **Missing Attribute Values:** None

Internet advertisements

Title of the Database: Internet advertisement

2. Sources:

(a) Creator & donor: Nicholas Kushmerick <nick@ucd.ie>
(c) Generated: April-July 1998

3. Past Usage:

N. Kushmerick (1999) "Learning to remove Internet advertisements", 3rd Int Conf Autonomous Agents.
Available at www.cs.ucd.ie/staff/nick/research/download/kushmerick-aa99.ps.gz.
Accuracy >97% using C4.5rules in predicting whether an image is an advertisement.

4. This dataset represents a set of possible advertisements on Internet pages. The features encode the geometry of the image (if available) as well as phrases occurring in the URL, the image's URL and alt text, the anchor text, and words occurring near the anchor text. The task is to predict whether an image is an advertisement ("ad") or not ("nonad").

5. Number of Instances: 3279 (2821 nonads, 458 ads)

6. Number of Attributes: 1558 (3 continous; others binary; this is the "STANDARD encoding" mentioned in the [Kushmerick, 99].)

One or more of the three continous features are missing in 28%
of the instances; missing values should be interpreted as "unknown".

7. See [Kushmerick, 99] for details of the attributes; in
".names" format:

Height: continuous. | Possibly missing
Width: continuous. | Possibly missing
Aratio: continuous. | Possibly missing
Local: 0, 1.
| 457 features from url terms, each of the form "url*term1+term2..."
| for example:
url*images+buttons: 0, 1.
...
| 495 features from origurl terms, in same form; for example:
origurl*labyrinth: 0, 1.
...
| 472 features from ancurl terms, in same form; for example:
ancurl*search+direct: 0, 1.
...
| 111 features from alt terms, in same form; for example:
alt*your: 0,1.
...
| 19 features from caption terms
Caption*and: 0, 1.
...

8. **Missing Attribute** Values: how many per each attribute?
28% of instances are missing some of the continous attributes.

9. Class Distribution: number of instances per class
2821 nonads, 458 ads.

Credit Approval

2. Sources: (confidential)
Submitted by quinlan@cs.su.oz.au

3. Past Usage:

See Quinlan,
* "Simplifying decision trees", Int J Man-Machine Studies 27, Dec 1987, pp. 221-234.
* "C4.5: Programs for Machine Learning", Morgan Kaufmann, Oct 1992

4. Relevant Information:

This file concerns credit card applications. All attribute names and values have been changed to meaningless symbols to protect confidentiality of the data.

This dataset is interesting because there is a good mix of attributes -- continuous, nominal with small numbers of values, and nominal with larger numbers of values. There are also a few missing values.

5. Number of Instances: 690

6. Number of Attributes: 15 + class attribute

7. Attribute Information:

A1: b, a.
A2: continuous.
A3: continuous.
A4: u, y, l, t.
A5: g, p, gg.
A6: c, d, cc, i, j, k, m, r, q, w, x, e, aa, ff.
A7: v, h, bb, j, n, z, dd, ff, o.

A8: continuous.
A9: t, f.
A10: t, f.
A11: continuous.
A12: t, f.
A13: g, p, s.
A14: continuous.
A15: continuous.
A16: +,- (class attribute)

8. Missing Attribute Values:

37 cases (5%) have one or more missing values. The missing values from particular attributes are:

A1: 12
A2: 12
A4: 6
A5: 6
A6: 9
A7: 9
A14: 13

9. Class Distribution

+: 307 (44.5%)
- : 383 (55.5%

Ozone later detection

1. Title: Ozone Level Detection

2. Source:

Kun Zhang
zhang.kun05 '@' gmail.com
Department of Computer Science,
Xavier University of Lousiana

Wei Fan wei.fan '@' gmail.com
IBM T.J.Watson Research

XiaoJing Yuan xyuan '@' uh.edu
Engineering Technology Department,
College of Technology, University of Houston

3. Past Usage:

Forecasting skewed biased stochastic ozone days: analyses, solutions and beyond, Knowledge and Information Systems, Vol. 14, No. 3, 2008. Discusses details about the dataset, its use as well as various experiments (both cross-validation and streaming) using many state-of-the-art methods.

A shorter version of the paper (does not contain some detailed experiments as the journal paper above) is in:
Forecasting Skewed Biased Stochastic Ozone Days: Analyses and Solutions. ICDM 2006: 753-764

4. Relevant Information:

The following are specifications for several most important attributes that are highly valued by Texas Commission on Environmental Quality (TCEQ). More details can be found in the two relevant papers.

-- O 3 - Local ozone peak prediction
-- Upwind - Upwind ozone background level
-- EmFactor - Precursor emissions related factor
-- Tmax - Maximum temperature in degrees F
-- Tb - Base temperature where net ozone production begins (50 F)
-- SRd - Solar radiation total for the day
-- WSa - Wind speed near sunrise (using 09-12 UTC forecast mode)
-- WSp - Wind speed mid-day (using 15-21 UTC forecast mode)

5. Number of Instances: 2536

6. Number of Attributes: 73

7. Attribute Information:

1, 0 | two classes 1: ozone day, 0: normal day

Appendix B:

Classifier Workflow Evaluation Questionnaire

The aim of this questionnaire is to evaluate the classifier workflow developed for classification projects in a single application domain. Before filling in the questionnaire please read carefully the classifier workflow together with the description of its phases provided in the attached document.

1. How familiar are you with machine learning and classification?
   Not Familiar ☐ Somewhat familiar ☐ Familiar ☐ Very familiar ☐

2. Is the classifier workflow understandable?
   Yes ☐ No ☐ Unsure ☐

Please state how much you agree or disagree with the following statements

3. The classifier workflow contains enough phases for machine learning and data mining classification experiments in a specific domain e.g. financial, health.
   Strongly agree☐ Agree ☐ Don't know ☐ Disagree ☐ Strongly disagree ☐

4. The second phase of the classifier workflow suggests a valid way for evaluating algorithms for machine learning and data mining classification experiments.
   Strongly agree☐ Agree ☐ Don't know ☐ Disagree ☐ Strongly disagree ☐

5. The suggested dataset thresholds in phase four are valid for performance estimation methods.
   Strongly agree☐ Agree ☐ Don't know ☐ Disagree ☐ Strongly disagree ☐

6. The suggested way for setting key parameters of classification techniques provided in phase six is valid for machine learning and data mining classification experiments

   Strongly agree☐ Agree ☐ Don't know ☐ Disagree ☐ Strongly disagree ☐

7. The classifier workflow can be applied in a real world classification projects.

   Strongly agree☐ Agree ☐ Don't know ☐ Disagree ☐ Strongly disagree ☐

8. What are the challenges facing the classifier workflow for its application in a real world problem?

9. Please use the following space to provide any other thoughts regarding the proposed workflow.

Bibliography

Alpaydin, E., 2004. *Introduction to Machine Learning*, Cambridge, Massachusetts, London, England: MIT Press.

Antony, J., 2003. *Design of Experiments for Engineers and Scientists*, Butterworth-Heinemann.

Baldi, P. & Brunak, S., 2001. *Bioinformatics*,

Bauer, E. & Kohavi, R., 1999. An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. *Machine Learning*, 36(1), 105-139.

Bengio, Y. & Grandvalet, Y., 2004. No Unbiased Estimator of the Variance of K-Fold Cross-Validation. *Journal of Machine Learning Research*, 5, 1089–1105.

Berg, K.E. & Latin, R.W., 2007. *Essentials of Research Methods in Health, Physical Education, Exercise*, Lippincott Williams & Wilkins.

Beri, G., *Marketing Research, 3e* Third., Tata McGraw-Hill.

Bernhard, B., 2002. Support Vector Machines and Kernel Methods: The new generation of learning machines. *AI Magazine*.

Berson, A., Smith, S.J. & Thearling, K., 1999. *Building Data Mining Applications for CRM*, Mc Graw-Hill.

Berthold, M. & Hand, D.J., 2003. *Intelligent Data Analysis*, Springer.

Breiman, L., 1996. Bagging predictors. *Machine Learning*, 24, 123--140.

Breiman, L., 2001. Random Forests.

Breiman, L., 2000. SOME INFINITY THEORY FOR PREDICTOR ENSEMBLES. *Technical Report 547, Statistics Dept. UCB*. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.24.7078 [Accessed August 7, 2008].

Breiman, L. et al., 1993. *Classification and Regression Trees*, Chapman & Hall Press.

Brunelli, R. & Poggio, T., 1993. Face recognition: features versus templates. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(10), 1042-1052.

Caelli, T. & Bischof, W.F., 1997. *Machine Learning and Image Interpretation*, New

York, NY, USA: Plenum Press.

Campbell, C., 2000. An Introduction to Kernel Methods. *Radial Basis Function Networks: Design and Applications*, 31-38.

Carter, C. & Catlett, J., 1987. Assessing Credit Card Applications Using Machine Learning. *IEEE Expert*, 2(3), 71-79.

Cherkasov, A., 2004. CodeProject: Creating Optical Character Recognition (OCR) applications using Neural Networks. Free source code and programming help. Available at: http://www.codeproject.com/KB/dotnet/simple_ocr.aspx?fid=15088&df=90&mpp=25&noise=3&sort=Position&view=Quick&select=2722077&fr=51 [Accessed September 28, 2008].

Collier, K. et al., 1998. A Perspective on Data Mining. *Northern Arizona University, July*.

Cormen, T.H. et al., 2003. *Introduction to Algorithms*, MIT Press.

Craven, M.W., 1996. Extracting Comprehensible Models from Trained Neural Networks.

CRISP-DM, 2000. CRISP-DM 1.0- Step by Step data mining guide.

Cunningham, P. & Delany, S.J., 2007. k-Nearest Neighbour Classifiers. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.100.1131 [Accessed August 5, 2008].

Demuth, J.E., 1999. *Basic Statistics and Pharmaceutical Statistical Applications*, Marcel Dekker.

Dietterich, T.G., 2000a. An experimental comparison of three methods for constructing ensembles of decision trees. *Bagging, boosting, and randomization. Machine Learning*, 40, 139--157.

Dietterich, T.G., 2000b. Ensemble methods in machine learning. , 1857, 1--15.

Dietterich, T.G., 1998. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithm. In MIT Press, pp. 1895-1923.

Esmeir, S. & Markovitch, S., 2006. When a Decision Tree Learner Has a Plenty of Time. In Cambridge, MA: AAAI Press, pp. 1597-1600.

Everitt, B., 1992. *The Analysis of Contingency Tables*, Chapman & Hall/CRC.

Fawcett, T., 2004. ROC Graphs: Notes and Practical Considerations for Data Mining Researchers.

Fayyad, U., Piatetsky-Shapiro, G. & Smyth, P., 1996. The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39(11), 27-34.

Feldens, M. et al., 1998. Towards a methodology for the discovery of useful knowledge combining data mining, data warehousing and visualization. *CONFERENCIA LATINOAMERICANA DE INFORMATICA (CLEI'98), XXIV*, 935-947.

Field, A.P., 2005. *Discovering Statistics Using SPSS*, SAGE.

Flynn, R.R., 1986. An Introduction to Information Science. In CRC Press.

Foxall, G., 2002. *Consumer Behaviour Analysis: a behavioural perspective*, Taylor&Francis.

Freund, Y. & Schapire, R.E., 1996. Experiments with a New Boosting Algorithm. *San Francisco*, 148–156.

Galindo, J. & Tamayo, P., 2000. Credit Risk Assessment Using Statistical and Machine Learning: Basic Methodology and Risk Modeling Applications. *Computational Economics*, 15(1), 107-143.

Gurney, K., 1997. *An Introduction to Neural Networks*, CRC Press.

Han, J. & Kamber, M., 2002. *Data Mining: Concepts and techniques*, Kauffman Press.

Hand, D.J., 2006. Classifier Technology and the Illusion of Progress. *Statistical Sciences*, Vol.21(No.1), 1-15.

Hastie, T., Tibshirani, R. & Friedman, J., 2001. *The Elements of Statistical Learning*,

Hinton, G.E. & Sejnowski, T.J., 1986. Learning and relearning in Boltzmann machines. *Mit Press Computational Models Of Cognition And Perception Series*, 282-317.

Hopfield, J.J., 1982. Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79(8), 2554-2558.

Hornberg, A. & NetLibrary, I., 2007. *Handbook of Machine Vision*, Wiley-VCH.

IBM, DB2 Universal Database. Available at: http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp?topic=/com.ibm.datatools.datamining.doc/c_maximum_tree_depth.html [Accessed August 25, 2008].

Joachims, T., 2002. *Learning to Classify Text Using Support Vector Machines*, Springer.

K. Narita & H. Kitagawa, 2006. Mining Frequent Itemsets from Noisy Data. In *Data Engineering Workshops, 2006. Proceedings. 22nd International Conference on*. p. x117.

Kawahara, M., 1999. Mining Association Algorithm with Threshold Based on ROC Analysis. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.105.6121 [Accessed August 14, 2008].

Kohavi, R., 1995. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In pp. 1137-1143. Available at: h t t p / / r o b o t i c s Stanford edu/"ronnyk.

Kohonen, T., 2001. *Self-Organizing Maps* Third., Springer.

Kononenko, I., 2001. Machine Learning for Medical Diagnosis: history, state of the art and perspective. *Artificial Intelligence in Medicine*, 23(1), 89-109.

Kostiantis, S., 2007. Supervised Machine Learning: A Review of Classification Techniques. *Informatica*, Vol.31, 249-268.

Langley, P. & Simon, H., 1995. Applications of machine Learning and Rule Induction. *Communications of the ACM*, 38(11), 55-64.

Lavrač, N. et al., 2003. *Knowledge Discovery in Databases*, Springer.

Liekens, A., 2007. Data mining musical profiles. Available at: http://images.google.com/imgres?imgurl=http://anthony.liekens.net/images/cluster12.png&imgrefurl=http://anthony.liekens.net/index.php/Computers/Data Mining [Accessed August 24, 2008].

Lin, C. & Lee, C., 1991. Neural-network-based fuzzy logic control and decision system. *Computers, IEEE Transactions on*, 40(12), 1320-1336.

Lin, C., 2006. Support Vector Machines.

Manning, C.D. & Schütze, H., 1999. *Foundations of statistical natural language processing*, MIT Press.

Matignon, R., Institute, S.A.S. & NetLibrary, I., 2007. *Data Mining Using SAS Enterprise Miner*, Wiley-Interscience.

McNemar, Q., 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2), 153.

Mehrotra, K., Mohan, C.K. & Ranka, S., 1997. *Elements of Artificial Neural Networks*, Cambridge, Massachusetts, London, England: MIT Press.

Mena, J., 1999. *Data Mining Your Website*, Digital Press.

Micheli-Tzanakou, E., 1999. *Supervised and Unsupervised Pattern Recognition*, CRC Press.

Mitchell, T., 1997. *Machine Learning*, MIT Press.

Mramor, D. & Zupan, J., 2008. Consumer credit scoring models with limited data.

Expert Systems with Applications. Available at: http://www.sciencedirect.com/science?_ob=ArticleURL&_udi=B6V03-4SRW12C-4&_user=10&_rdoc=1&_fmt=&_orig=search&_sort=d&view=c&_version=1&_urlVersion=0&_userid=10&md5=e441956f977468126532debf047d54a4 [Accessed August 24, 2008].

Nelles, O., 2001. *Nonlinear System Identification*,

Osborn, C.E., 2005. *Statistical Applications For Health Information Management*, Jones & Bartlett Publishers.

Osuna, E., Freund, R. & Girosit, F., 1997. Training support vector machines: an application to face detection. In pp. 130-136.

Oxford English Dictionary, 1989. Oxford English Dictionary. In Oxford University Press.

Prechelt, L., 1996. A quantitative study of experimental evaluations of neural network learning algorithms: Current research practice. *Neural Networks*, 9(3), 457-462.

Provost, F., Fawcett, T. & Kohavi, R., 1999. The Case Against Accuracy Estimation for Comparing Induction Algorithms. In San Diego, California, United States, pp. 155-164.

Quinlan, J., 1993. *C4.5: Programs for Machine Learning*, San Mateo, CA: Morgan Kauffman.

Rajashekaran, S., Pai, G.A.V. & Vijayalksmi, G., 2004. *Neural Networks, Fuzzy Logic and Genetic Algorithms*,

Raynor, W., 1999. *The International Dictionary of Artificial Intelligence*, Chicago and London: Fitzroy Dearborn Publishers.

Rice, S.V., Nagy, G. & Nartker, T.A., 1999. *Optical Character Recognition*,

Roiger, R. & Geatz, M., 2002. *Data Mining: A tutorial based primer*, Addison Wesley.

Rokach, L. & Maimon, O., 2005. Top-down induction of decision trees classifiers - a survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 35(4), 476-487.

Romesburg, C., 2004. *Cluster Analysis for Researchers*, Lulu. Com.

Safavian, S.R. & Landgrebe, D., 1991. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3), 660-673.

Salzberg, S.L., 1999. On Comparing Classifiers: A Critique of Current Research and Methods. In Boston: Kluwer Academic Publishers, pp. 1-12.

SAS Institute, S.A.S., 2003. *Data Mining Using Sas Enterprise Miner* second., SAS Publishing.

Sebastiani, F., 2002. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1), 1-47.

Siddiqi, J. et al., 2008. Automated Diagnosis System to Support Colon Cancer Treatment: MATCH. In *Information Technology: New Generations, 2008. ITNG 2008. Fifth International Conference on*. pp. 201-205.

Stolean, R. et al., 2007. Concerning the Potential of Evolutionary Support Vector Machines. In IEEE, pp. 1436-1443.

Tang, Y. et al., 2004. Granular Support Vector Machines for Medical Binary Classification Problems. In pp. 73-78.

Winkler, J., Niranjan, M. & Lawrence, N., 2005. *Deterministic and Statistical Methods in Machine Learning*, Birkhauser.

Wirth, R. & Hipp, J., 2000. CRISP-DM: Towards a Standard Process Model for Data Mining. *Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining*, 29–39.

Witten, I.H. & Frank, E., 2000. *Data Mining*, Morgan Kauffman.

Xiao-Dong Liu, Chun-Yi Shi & Xue-Dao Gu, 2005. A boosting method to detect noisy data. In *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*. pp. 2015-2020 Vol. 4.

Zhang, N. & Lu, W., 2007. An Efficient Data Preprocessing Method for Mining Customer Survey Data. In *Industrial Informatics, 2007 5th IEEE International Conference on*. pp. 573-578.

Zhao, H. & Sinha, A., 2005. An Efficient Algorithm for Generating Generalized Decision Forests. *Systems, Man and Cybernetics, Part A, IEEE Transactions on*, 35(5), 754-762.

Zhong, N. & Liu, J., 2004. *Intelligent Technologies for Information Analysis*, Springer.

Zhou, L., Ooi, B.C. & Meng, X., 2005. Database Systems for Advanced Applications. In *Proceedings Database Systems for Advanced Applications: 10th International Conference*. Springer.

Ziarko, W. & Yao, Y., 2001. *Rough Sets and Current Trends in Computing*,